

Voice Control With The VeeR VRbot Module

SERVO

FOR THE ROBOT INNOVATOR
www.servomagazine.com

MAGAZINE
May 2010

STAIR - "CONTEXT" BASED OBJECT DETECTION

Adding
The **3D**
Component

◆ XBee Pro Wireless Sensor Network

- Monitor sensors
- Display status on an LCD
- Activate relays
- Flash status LEDs

◆ Preparing For A Robot Competition

Insight and tips from a combat veteran

U.S. \$5.50 CANADA \$7.00

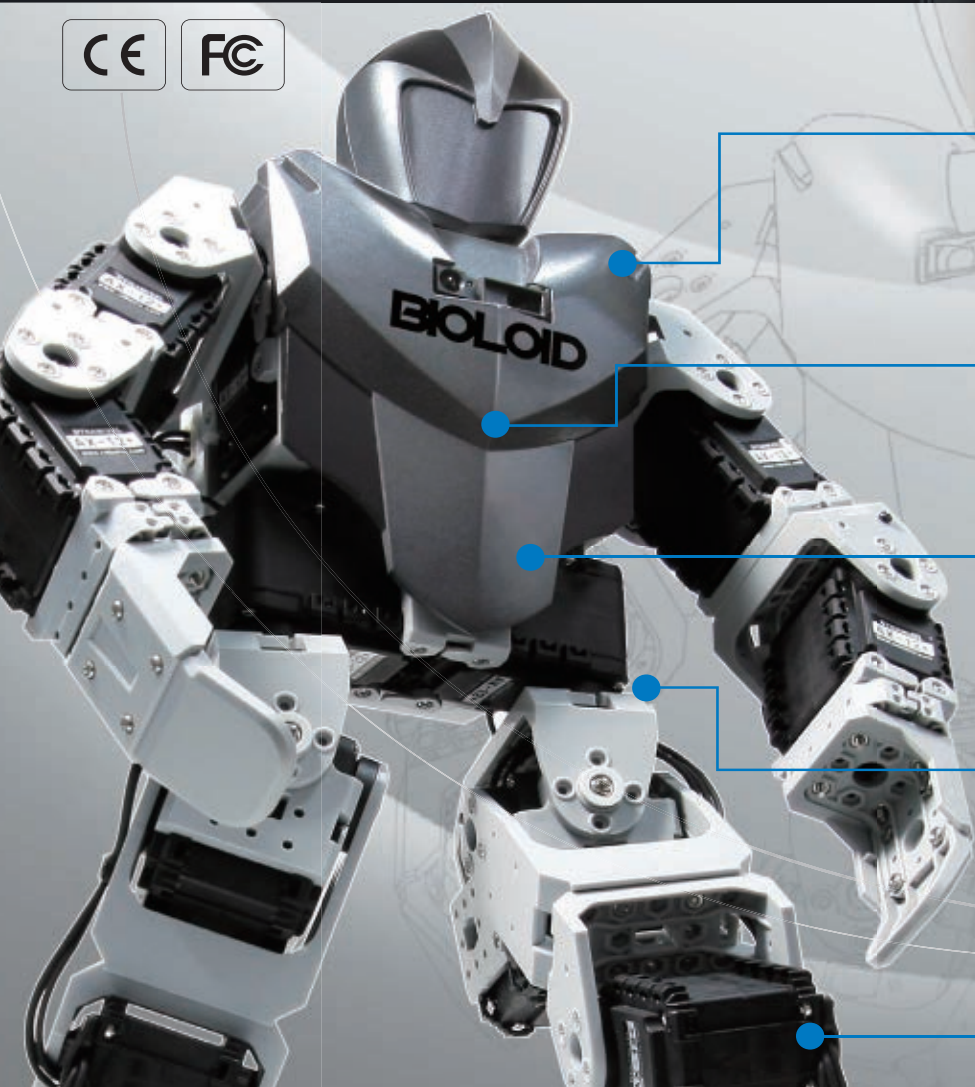


BIOLOID

PREMIUM Kit

Authorized kit by **ROBO ONE**

Possible to build diverse type of robots and program by yourself
Optimized material for robot education class



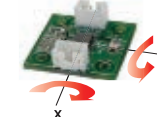
CM-510
Main Controller



RoboPlus



Gyro Sensor
Walk Balancing



DMS
Distance Measurement



Humanoid Skin



RC-100
Remote Controller



Li-Po Battery
11.1V 1000mA



USB2Dynamixel



DYNAMIXEL AX-12+
18 DOF Actuators



IR Sensor
Object Detection



DYNAMIXEL

High-performance robot exclusive actuator

ROBOTIS
www.robotis.com

NEW AX-12+ / AX-18F



	AX-12+	AX-18F
Weight(g) / (oz)	53.5(g) / 1.88(oz)	54.5(g) / 1.92(oz)
Dimension(mm) / (inch)	32×50.1×40(mm) 1.25×1.97×1.57(inch)	32×50.1×40(mm) 1.25×1.97×1.57(inch)
Gear Ratio(material)	1 : 254(enpla)	1 : 254(enpla)
Operation Voltage(V)	9~12	9~12
Holding Torque(kgf·cm)	15 at 12V / 1.5A	18 at 12V / 2.2A
No load speed(RPM)	59	97
Network Interface	TTL	TTL
Position Sensor(Resolution)	Potentiometer(300°/1024)	Potentiometer(300°/1024)
Motor	Cored Motor	Coreless Motor

NEW RX-64



	RX-64
Weight(g) / (oz)	125(g) / 4.4(oz)
Dimension(mm) / (inch)	40.1×61.3×45.8(mm) 1.57×2.41×1.8(inch)
Gear Ratio(material)	1 : 200(metal)
Operation Voltage(V)	12~18.5
Holding Torque(kgf·cm)	52 at 18.5V / 2.6A
No load speed(RPM)	64
Network Interface	RS-485
Position Sensor(Resolution)	Potentiometer(300°/1024)
Motor	Maxon Motor

NEW RX-24F/ RX-28



	RX-24F	RX-28
Weight(g) / (oz)	67(g) / 2.36(oz)	72(g) / 2.53(oz)
Dimension(mm) / (inch)	35.5×50.8×41.8(mm) 1.39×2×1.64(inch)	35.5×50.8×41.8(mm) 1.39×2×1.64(inch)
Gear Ratio(material)	1 : 193(metal)	1 : 193(metal)
Operation Voltage(V)	9~12	12~18.5
Holding Torque(kgf·cm)	26 at 12V / 2.4A	37 at 18.5V / 1.9A
No load speed(RPM)	126	67
Network Interface	RS-485	RS-485
Position Sensor(Resolution)	Potentiometer(300°/1024)	Potentiometer(300°/1024)
Motor	Coreless Motor	Maxon Motor

NEW EX-106+



	EX-106+
Weight(g) / (oz)	154(g) / 5.43(oz)
Dimension(mm) / (inch)	40.1×65.3×50.1(mm) 1.57×2.57×1.97(inch)
Gear Ratio(material)	1 : 184(metal)
Operation Voltage(V)	12~18.5
Holding Torque(kgf·cm)	107 at 18.5V / 7A
No load speed(RPM)	91
Network Interface	RS-485
Position Sensor(Resolution)	Magnetic encoder(251°/4096)
Motor	Maxon Motor

The Future of Servo Control is Calling...

ServoCenter™

Features

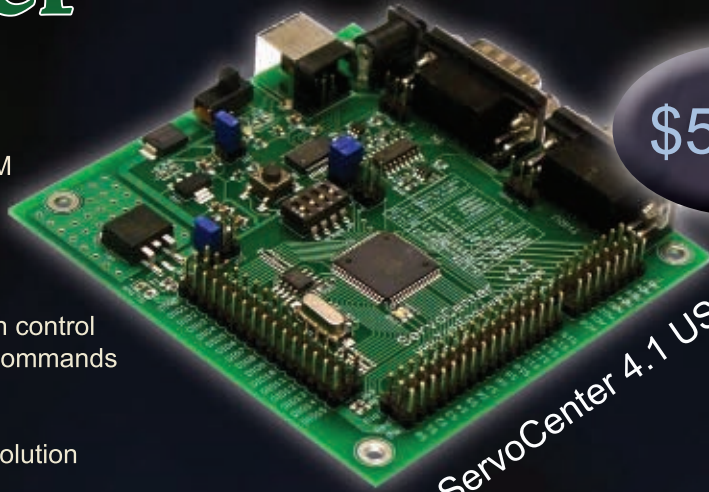
- USB, RS232, and TTL serial support
- 16 servos, 16 digital I/O, 8 analog inputs
- Built-in SC-BASIC Sequencer with EEPROM
- Sequencer allows stand-alone operation
- 64 scene presets stored in EEPROM
- Presets instantly loaded or cross-faded
- Built-in configurable smoothing algorithm
- Independent, simultaneous speed & position control
- Scaled, percentage, and group movement commands
- Timed movement commands
- Max, min, & startup position settings
- Ultra-precise 0.05425μS jitter-free pulse resolution

Flexibility

- User upgradeable firmware
- Upload your own firmware with bootloader
- Watchdog timer for failsafe operation
- Over-current, over-temperature, polarity protection
- Internal regulator, external power, or battery power options
- Supports 4.8/6.0V regulated servo supply voltages at 5 Amps
- Each digital I/O and analog input has supply pins
- Direct serial, activeX control, or Win32 DLL communication
- Programming examples in 10+ languages
- Windows, Linux, Mac OSX compatible

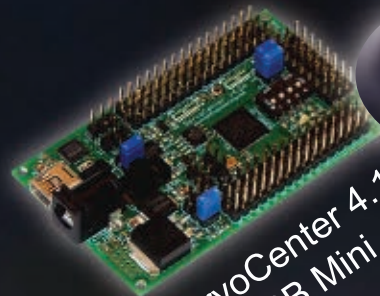
Free Control Panel Software

- Easy editing and configuration of servo settings
- Configure and set up digital I/O & ADC settings
- Edit scene presets
- Program the SC-BASIC sequencer
- Upload and run your SC-BASIC programs
- Debug & communicate with terminal window



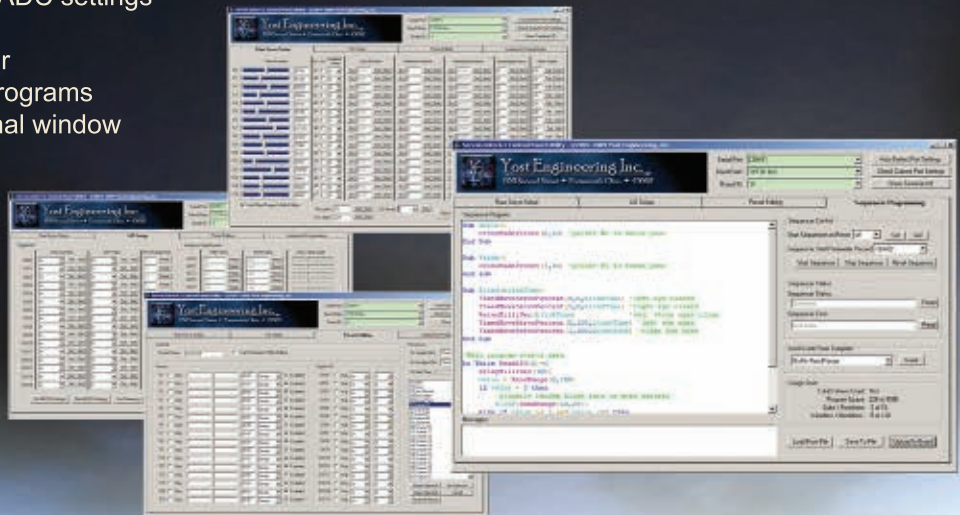
\$56.99

ServoCenter 4.1 USB



\$59.99

ServoCenter 4.1
USB Mini



www.Servo-Center.com

Yost Engineering Inc.

SERVO

MAGAZINE

05.2010

VOL. 8 NO. 5

The Combat Zone...

Features

- 26 BUILD REPORT:**
Reversing the Trend
- 27 MANUFACTURING:**
DIY CNC for CAD/CAM
- 30 PARTS IS PARTS:**
Kitbots Rolls Out B-16
Gearmotor Mounts
- 31 RioBotz Combot Tutorial:**
LaunchBots
- 34 Combat Zone's**
Greatest Hits

Events

- 35 Results/Upcoming**
Events
- 35 EVENT REPORT:**
Motorama 2010



PAGE 78

Columns

- 08 Robytes**
by Jeff Eckert
Stimulating Robot Tidbits
- 10 GeerHead**
by David Geer
Hot Dog! Robot serves up wieners!
- 13 Ask Mr. Roboto**
by Dennis Clark
Your Problems Solved Here
- 72 Twin Tweaks**
by Bryce and Evan Woolley
RoboNova — Come Here!
I Want to See You!
- 78 Then and Now**
by Tom Carroll
What Does a Robot Look Like?



PAGE 28



PAGE 31

Departments

- 06 Mind/Iron**
- 16 Events Calendar**
- 18 New Products**
- 21 Showcase**
- 22 Bots in Brief**
- 64 SERVO Webstore**
- 81 Robo-Links**
- 81 Advertiser's Index**

SERVO Magazine (ISSN 1546-0592/CDN Pub Agree#40702530) is published monthly for \$24.95 per year by T & L Publications, Inc., 430 Princeland Court, Corona, CA 92879. PERIODICALS POSTAGE PAID AT CORONA, CA AND AT ADDITIONAL ENTRY MAILING OFFICES. POSTMASTER: Send address changes to **SERVO Magazine, P.O. Box 15277, North Hollywood, CA 91615** or Station A, P.O. Box 54, Windsor ON N9A 6J5; cpcreturns@servomagazine.com

In This Issue ...

38 Taking the STAIRs to Advance Robot Development

by *Jeremy Kerfs*

Researchers at Stanford and Cornell are pioneering new methods of teaching robots to interact more fluently with their surroundings using the STanford Artificial Intelligence Robot (STAIR) as the base.

42 How to Prepare For — and Maybe Even Win — a Robot Competition

by *Pete Smith*

Get some insights and helpful tips from a veteran of robot combat.

49 So, You Want to Build a ComBot — Part 2

by *Greg Intermaggio*

This time, we'll put everything together to turn that bucket o' bolts into a working ComBot.

54 Go XBee-PRO With Your Robot Control

by *Fred Eady*

Learn how to monitor systems, display status via an LCD, activate relays, drive solid-state relays, and flash status LEDs, all via the top of a tiny wire antenna.

61 GPS Navigation — Part 3

by *Chris Savage*

Our navigation code will get a bit more complex, so this month the focus will be on how the individual sections work.

68 Using a VEX Controller for Electronic Experiments

by *Daniel Ramirez*

Build a DIY numeric LED display that's bright and easy to read, while at the same time conserves battery power and precious I/O pins.

PAGE 42



PAGE 49



PAGE 61



Mind / Iron

by Bryan Bergeron, Editor



Mechatronic Prosthetics

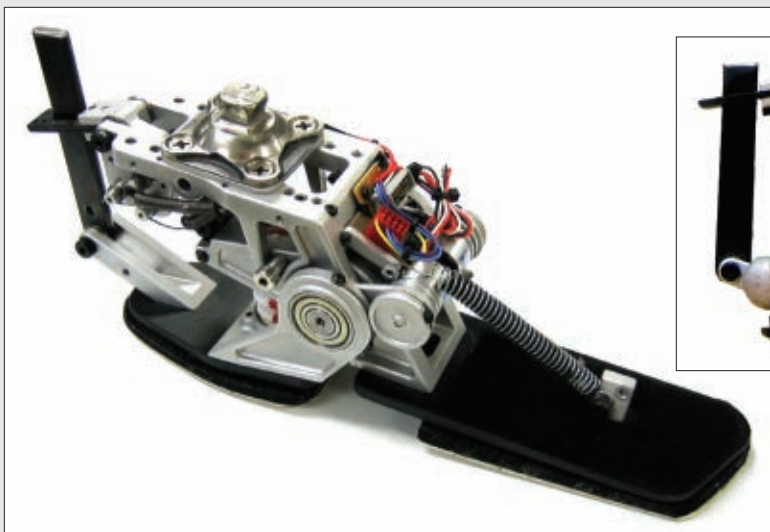
Semi-autonomous robot toys and planetary rovers are exciting technologies, but the real promise of practical robotics is in alleviating human pain, suffering, and disability. Accidents, arthritis, and the normal decline in function with aging are a few reasons companies and universities in the US, Japan, and the EU are developing mechatronic prosthetics. You've probably heard about the various robotic prosthetic arm projects under development over the past several years. These arms have showcased advances in both myoelectric (control signals from muscles) and neurologic (control signals from the nervous system) control.

One of the latest examples of robotic or mechatronic prosthetic technology is an energy-recycling artificial foot developed by Art Kuo and Steve Collins at the University of Michigan. The foot (shown in the accompanying photos) uses the otherwise wasted energy of the normal

with walking with a biological foot. The new design drops the energy penalty by about 10%. The goal of most researchers in this area is to match and then exceed the mechanical efficiency of biological feet.

According to the inventors, the foot isn't the first to recycle the energy of walking. All prosthetic feet based on a simple spring design store energy on impact and then release it. The problem with this generic spring design is that the energy is released as soon as the tension on the spring is lessened. However, this timing generally has no relationship with the energy required during push-off. Using onboard processing and a built-in battery, the foot determines exactly when to release the pent-up energy in the spring to maximally supplement push-off.

I liken the operation of the device to that of a mousetrap. The impact of the heel on the ground tenses the spring, and it latches when the spring is sufficiently tensed. Similarly, to set a mouse trap you have to



heel strike to enhance the ankle push-off. The foot uses a microcontroller and two micro-motors to release the spring and reset the mechanism. An internal battery provides about 0.8W to power the assembly. It's not clear from the literature how long the battery lasts between charges or replacement.

This mechatronic foot is significant for amputees because it promises to make the effort of walking less arduous. Conventional foot prosthetics require the wearer to expend about 24% more energy to walk — compared



tense the spring and then latch it to a trip mechanism. When a mouse touches the trip mechanism, the spring is suddenly released with devastating consequences.

It's the same with the foot. The difference is that the burst of released energy is used to launch the person forward. In addition, the foot automatically resets itself when the heel hits the ground. The advantage of this design over previous designs is that the foot is self-contained. There are no cables to external batteries or controllers. I can envision a time when the efficiency of the design is improved to the point that the foot would generate enough energy to power the onboard electronics without the need for a battery.

The ancillary applications of this technology should

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct., Corona, CA 92879-1300
(951) 371-8497
FAX (951) 371-3052
Webstore Only 1-800-783-4624
www.servomagazine.com

Subscriptions
Toll Free 1-877-525-2539
Outside US 1-818-487-4545
P.O. Box 15277, N. Hollywood, CA 91615

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

**ASSOCIATE PUBLISHER/
VP OF SALES/MARKETING**
Robin Lemieux
display@servomagazine.com

EDITOR
Bryan Bergeron
techedit-servo@yahoo.com

CONTRIBUTING EDITORS
Jeff Eckert
Tom Carroll
Dennis Clark
Fred Eady
Gregory Intermaggio
Jeremy Kerfs
Pete Smith
Bryce Woolley
Chris Olin

Jenn Eckert
David Geer
R. Steven Rainwater
Kevin Berry
Chris Savage
Daniel Ramirez
Jerome Miles
Evan Woolley

CIRCULATION DIRECTOR
Tracy Kerley
subscribe@servomagazine.com

**MARKETING COORDINATOR
WEBSTORE**
Brian Kirkpatrick
sales@servomagazine.com

WEB CONTENT
Michael Kaudze
website@servomagazine.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION/GRAPHICS
Shannon Christensen

Copyright 2010 by
T & L Publications, Inc.
All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *SERVO Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *SERVO*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *SERVO*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: **430 Princeland Court, Corona, CA 92879.**

Printed in the USA on SFI & FSC stock.



be obvious – it's a means of reducing the energy requirements of biped robots. I imagine the technique could be applied to any simple biped robot such as the Parallax Penguin (www.parallax.com) or the Lynxmotion Brat (www.lynxmotion.com), or a more complex robot such as a Kondo KHR humanoid robot (www.kondo-robot.com). In each case, the feet would need to be modified significantly because current

production models don't use spring feet. If you're up for the challenge, you should read the original report that details the foot's operation. "Recycling Energy to Restore Impaired Ankle Function during Human Walking" by Steven Collins and Arthur Kuo (*Plos ONE*, Feb. 2010, Vol. 5, Issue 3, e9307) is available at www.plosone.org. A PDF of the article is available for free download. **SV**

Special thanks to Steven Collins for sharing his photos with us.

Coming Soon To A SERVO Tankbot Near You! Meet our new columnist, Calvin Turzillo



Well, well, well ... What do we have here? Yet another hip youngster sticking their nose into the affairs of robotic aficionados? Okay, fine, so I'm not *that* "hip," but I do know my way around a circuit or two. I suppose you could say that I am the latest recruit into the *SERVO Magazine* army, here to entertain and inform the masses. My purpose is to explore strange new sensors; to seek out new code and new implementations; to boldly go where no Tankbot has gone before! Yeah ... that was just a little lame on my part, but I promise you that I'll keep my articles light and interesting while trying to cater to audiences both young and old.

Now you may be asking, "Who does this kid think he is? What does he know about robots?!" I'm glad you inquired imaginary person! I currently hold a Bachelors degree in Electrical Engineering and a Masters degree in Space Systems Engineering. In both degrees, I focused on mechatronics, control systems, and long range and long duration robotic operations. While my later studies were mainly in radioisotope thermoelectric generator

powered interplanetary robotic missions (a.k.a., little green glowing robonauts), I have built various robotic arms and RC helicopters in my spare time for fun. In my professional career, I have been the lead electrical engineer for two space shuttles, one of the lead Ares I-X vehicle integration engineers, and I am currently working on pushing man beyond low earth orbit as part of the Constellation program. I'm also 26 years old, a virgo, and enjoy long walks on the... Oh wait... That's for a different "article"...

So, here comes the "ask not what your Tankbot can do for you; ask what you can do for your Tankbot" moment. While I have plenty of ideas about what to do with the Tankbot kit, I want to know what you — the kind, good looking, and incredibly intelligent reader — would like to see (flattery gets you everywhere, right?). Please send your thoughts, ideas, criticism, or boundless praise to DrTankBot@gmail.com and I will do my best to incorporate everyone's input.

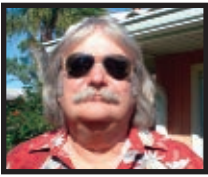
I'm looking forward to starting this new endeavor with my first full length article in a month or two, so please send me those emails!

Allons-y!

Order your Tankbot Kit now from the
SERVO Webstore!

See Page 66 in this issue
or go online at

www.servomagazine.com



Robytes

by Jeff and Jenn Eckert

Robot Theme Park Underway



Overview of Korea's Robot Land, scheduled for a 2013 competition.

The South Koreans have long viewed robotics as a key element in the country's economic growth, and their enthusiasm is no longer limited to industrial and service machinery. The latest symptom of their bot fever will soon manifest itself as "the world's first robot theme park" — Robot Land — which was recently assigned a spot inside the Incheon Free Economic Zone. According to the Ministry of Knowledge Economy, the park will cost a little less than 800 billion won (about \$562 million) which seems like a real deal considering that it will cover an area of about five million square feet and include the Robot Experience Center, a bot battle stadium, a robot museum, and various other theme-related attractions ... plus a water park, roller coaster, cyber zoo, some hotels, and the corporate and R&D facilities.

Over here, I'm not sure you can even put up a Motel 6 for only \$562 million. These clever people are doing it with the government putting up less than 15 percent of the cost, with the lion's share coming from private investors. Construction will begin this year and be completed by 2013, with some parts being open to the public as early as next year. In theory, the official website is www.robotland.or.kr/eng/index.php, but it was malfunctioning as of this writing.

UAV Altered for Disasters

If you've ever wondered if anyone is preparing for Armageddon, rest

The guts of a Yamaha RMAX UAV.



assured. It was recently revealed that a team of students at Virginia Tech's Unmanned Systems Lab (www.me.vt.edu/unmanned/) is reengineering a remote-controlled Yamaha RMAX UAV for fully autonomous operation, with the overall goal of providing surveillance of nuclear disasters. The six ft, 200 lb helicopter has been modified to inspect damage caused by a standard nuclear blast or a dirty bomb, measuring radiation levels and generating photos and maps of all the carnage and destruction. It even has its own special flight control software that will direct it to radioactive sources. The payload of plug-and-play instruments fits under the fuselage, and one configuration includes a small tethered bot that can venture out to collect samples either in chunks or by vacuuming up some dirt. Various packages include a stereo-camera system, night-vision equipment, and anti-smoke and fog vision capabilities. Plans are to have a fleet of them in mission-ready form within three years. In the meantime, as Eric Idle has advised, "Always look on the bright side of life."

Move Aside, HoverRound®

Lord knows that no one begrudges extremely aged and morbidly obese people their scooters and power chairs. Sure, they clog up the aisles at Wal-Mart, running over your toes and knocking things off the shelves as they ring their little bells



The RoboCar®.

and search for the cat food aisle. But hey, if not for the grace of God, and all that. But it looks like Japan's ZMP, Inc. (www.zmp.co.jp/), has extended the concept beyond the merely annoying. Reportedly, the company is taking orders for a single-seat electrical vehicle that is capable of autonomous operation. According to a company rep, "We are aiming to achieve a safe and comfortable (vehicle) for personal mobility in a low-carbon society and a new transportation system."

Among RoboCar's intended applications are transport of patients and medical equipment, and short-range transportation for the elderly — both within confined spaces. To do the job, it's equipped with a laser range finder, a GPS, a stereo camera, and other navigation devices. In such cases, the autonomous feature seems like a good thing; the rider won't have to remember where the lab is or

why he's going there. But isn't it only a matter of time before Grandpa gets bored, switches the vehicle into manual mode, and heads for the nearest bar? Do we really want these things on the highway?

In any event, the RoboCar measures only 2480 x 1280 x 1370 mm (98 x 50 x 54 in), weighs in at 96 kg (212 lb), and — driven by a brushless DC motor — zips along at up to 60 km/h (37 mph). With a dual battery pack, it will go up to 60 km between charges. On the negative side, it's probably the only thing on the road that can't survive a collision with a SmartCar, so let's hope Gramps has his life insurance paid up.

Cyber Band Driven by Wiimotes

Mixing robotic technology, musical algorithms, and a bit of stagecraft is Jazari, a three-piece drum band consisting of two percussion instruments and their inventor, Patrick Flanagan. Although the two mechanical members are capable of operating on their own, Flanagan has added the human element for greater improvisational capabilities and more visual interest. In other words, audiences tend to think the drum thing is sort of boring without any accompanying movement, so "Patrick plays several roles: He improvises on one or two machines, controlling every note that they play; he triggers and loops material and shapes its overall character; and he interacts with his machines by playing one and allowing another to improvise a response of its own." For a video and other details, visit jazarimusic.com. The band, incidentally, is named after Abu al-Jazari (1136–1206), thought to be the inventor of the first musical automaton, among other things.



Jazari, a three-piece robotic drum circle.

Bot Brings Brew

Nothing goes together more naturally than beer and college, so Dr. Stephen Prior — with some students at the U.K.'s Middlesex University Product Design and Engineering Department (www.mdx.ac.uk/) — invented the perfect

study aid: a bot that pours your beer for you. It's basically a beer keg on a wheeled platform that can be programmed to move along a specific route. All you have to do is wave your hand over a sensor to stop it, stick a glass under the tap, and allow the beerbot to pour you a pint before heading for the next customer. Prior and his team are thinking about producing them commercially but surmise that each one would cost about \$2,250, so they're looking into cost-reduction ideas before pursuing it. (They might also want to test it with *cold* beer before trying to sell it outside the U.K.) In the meantime, you'll just have to suffer being served by a Hooters girl or a facsimile thereof.



Middlesex University's beer bot.

Report Outlines Spacebot Plans

If you happen to be a NASA fan and want to know what the agency is planning (including in the field of robotics), you might want to take a look at a report issued late last year by the Augustine Commission called "Seeking a Human

Spaceflight Program Worthy of a Great Nation." An interesting section is 9.6 Managing the Balance of Human and Robotic Spaceflight. I located a copy of the 157-page document which you can download by pointing your browser to www.jkeckert.com/freedownloads/FinalReport.pdf. **SV**

The Augustine Report on human spaceflight.





GEARHEAD

by David Geer

Contact the author at geercom@windstream.net

Hot Dog! Robot serves up wieners!

If you want people to pay attention to your technology, blend it with something that lots of people already love. In this case, that turned out to be the all-American hot dog!

Hot Dog Handler Inception

The Rochester Institute of Technology (RIT) robotics club was brainstorming one day on how to demonstrate the skills of the members. Suddenly, the perfect plan materialized in the form of a robotic chef that would help to feed the hungry RIT masses. Thus, the hot dog handler was born.

The hot dog robot turned out to be a good way to teach the students a number of new capabilities in manufacturing engineering, according to Kevin Laperriere and Dustin D'Angelo, undergraduates at RIT. "We learned machining skills, pneumatics, electronics, programming, fixturing, and how to control a system," commented D'Angelo and Laperriere.

The students were also looking for ways to interest youngsters in the manufacturing/engineering trade. "The hot dog robot proved to be very eye catching for children of all ages," Laperriere and D'Angelo smiled.

Hot dog vending robot with Adept robot arm serves dogs to a hungry crowd, assisted by student roboticists.

Design

In 2008, the roboticists first adapted an Adept robotic arm (an Adept Cobra 600) for picking and placing objects on a wooden bench. Next, they moved it around the campus and tried it at different locations.

Later (in 2009), the student roboticists designed add-on fixtures for the robot that could hold the buns and hot dogs, and manipulate them. This design process was done in Autodesk Inventor — a 3D manufacturing design and digital prototyping software package.



The fixtures were constructed of polyethylene (an FDA approved plastic for food service) and machined in the robotics department's machine shop. The bun fixture holds eight hot dog buns and the hot dog fixture holds an equal number of wieners (unlike the mismatched ratio of hot dogs to buns in typical grocery store packaging!).

"The hot dog fixture is fitted with an optical sensor which sends a visual signal via a status light on top to signal that the machine had to be reloaded with dogs and buns," explained D'Angelo and Laperriere.

"The coolest fixture on the robot is probably the ketchup and mustard pump. This fixture holds two pneumatic cylinders that push down on the pumpers to dispense the condiments."

The condiment pumping fixture is made of Bosch tubing pulled from scrap and designed "on-the-fly," machined, and assembled to the robot in about two hours' time. "We used trial and error to determine the timing and air pressure necessary to dispense the right amount of condiments onto the dogs."

Arm Choice and Other Parts

The team chose the Adept Cobra 600 because of its size and also because the Adept robot is tried and true, used by many in industry over several years. "The work envelope is sufficient to do projects such as the hot dog assembly line and

other light manufacturing projects we may choose to do," the roboticists explained. The arm is also just small enough so the team can store the robot's computer under the table it sits on. The entire cell can be moved on casters because of the robot's light weight. Other parts include two Norgren RLD06A SAN A400 single-acting, spring actuated pneumatic cylinders. "We bought these from McMaster-Carr to apply a force onto the condiment pumping fixture."

The robot employs a double-acting cylinder used to deliver plates from their storage space behind the cell glass to each customer. This is an RHINC base slide powered by an SMC NCGCN25-1000 air cylinder that is used to transfer each serving plate from one side of the cell to the other so the robot can place the hot dog and bun onto the plate. The slide is also used to maneuver the hot dog under the condiments so they can pump across the whole dog. A conveyor brings the plate to the "home" position so it can be pushed out of the cell.

The hot dog robot uses a five-port two-position solenoid activated valve from SMC to control all of the pneumatics on the robot — the condiment pumps, the pneumatic conveyor, and the cylinder.

The robot alerts its operators when the fixtures are empty by shining a 24 VDC 3 light. The light is red when the fixtures are empty and green when they are full.



Don't those dogs look good!

Robot Assembly

The robot is constructed from Bosch aluminum framing, a wooden table, four casters, plastic, and lexan safety glass. The work cell (or work area) is about 30 x 60 x 72 inches. The aluminum is bolted to the table on the top and bottom of the wood table all the way around the cell area. The aluminum uprights are connected to the table and casters using special Bosch hardware designed for this tubing.

Some of the framing was modified to build stands for the fixtures and "tapped" with UNC holes. "We drilled through the table and bolted the aluminum to the table. Then, we used set screws to hold dowel pins in place so we could consistently change over the fixtures and keep them in their appropriate locations," explained the roboticists.

The cell itself has two cabinet doors also made up of Bosch tubing and hardware. The main CPU for the robot was put inside of the cabinet, along with the electrical and pneumatic control panel. All the wiring and hoses were channeled under the robot arm.

Robot Intelligence

The hot dog robot is programmed in V+ code. All the I/Os (eight of them) are controlled via the code which executes one line at a time. "We used timing delays in the code and manually adjusted the air pressure using flow control valves prior to production runs in order to make sure all of the robot's components work right," commented the robotics students.

"Unfortunately, every time the robot is moved to a different location, the pressure from the source changes and all of the controls onboard the robot need to be adjusted to compensate for the changes in the supply source," D'Angelo and Laperriere continued.

The V+ programming has the flexibility to write subprograms and execute various programs depending on what the user selects from the Human-Machine-Interface (HMI) or from the touch screen. "For example, if a customer selects mustard on the HMI, the robot will execute the main program and then based on the input from the HMI, run the subprogram that puts the mustard on the hot dog."

Onboard communications exist only among the eight I/O configuration that comes with the Adept Cobra 600 robot arm. External communication is limited and exists between the HMI that talks to the robot via a nine-pin serial port. The touch screen display is by Touchsystems.

Its buttons are programmed in Visual Basic. When a button is pressed/touched, the VB code sets up the correct inputs required for V+ on the robot.

Here's the series of events the robot goes through to serve up a hot dog:

- A. Customer enters their order on the HMI touch screen.
- B. A variable is created from the user's input.
- C. The variable tells the programming what subprograms to activate.
- D. The software executes the appropriate order for the steps in sequence, making decisions based on if/or/and functions.
- E. The positions the robot needs to move into to fulfill the order are saved as x, y, and z coordinates.
- F. The pneumatics operate the robot via the I/Os.
- G. The solenoids activate.
- H. The cylinders extend and retract accordingly.

Robot Evolutions

Each year, the students re-build and improve on the robot. The current model (2009-2010) has improved fixtures to better hold the buns and hot dogs. The fixtures are what are known as Kanban systems which allow the roboticists to prepare other fixtures and put them in the same position as the previous fixtures.

This system includes the optical sensor that informs the operators when the hot dogs or buns run out.

Last Call

One might forecast that if these dogs are not hot, the roboticists might find their geese cooked! Fortunately, the machine is only tasked with vending the edible delights. Human hands stay in control of the grilling. **SV**

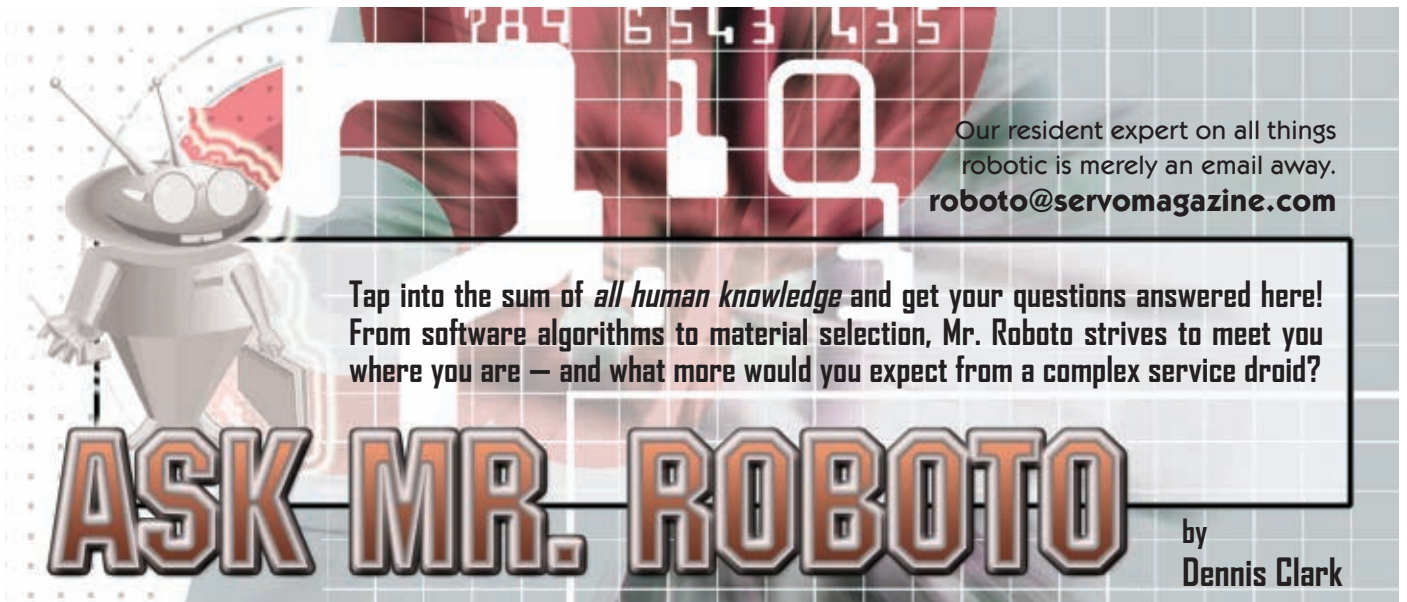
Resources

Rochester Institute of Technology
www.rit.edu

RIT Robotics Club
<http://mdrc.rit.edu>

RIT Robotics Lab
www.rit.edu/cast/mmetps/labs/robotics.php

RIT hot dog assembler; dispenser robot video
www.youtube.com/watch?v=enmuwG5rOGA



Last month, I covered how to install `avr-gcc`, `Eclipse`, and supporting code for playing with the Atmel ATMEGA AVR microcontrollers. I got through the full development environment up to writing our first program and compiling it. This month, I'm going to cover configuring `avrdude` to program an ATMEGA328, how to set `avrdude` up in `Eclipse` as a tool you can use with the press of an IDE button, and how to find where Ubuntu put your USB/serial converter so that you can tell `avrdude` about it.

Using Linux to develop on the Atmel AVR microcontroller – continued.

When last we met, we'd just written and compiled a "Hello World" embedded program that blinks an LED. I hope everyone got this to compile because this month we're going to program an ATMEGA328 to blink that LED. Let's get started.

Install AVR programming hardware, and finding the `dev` file.

The `dev` file you ask? Yup. Linux creates a hook to system drivers that talk to custom and common hardware in a directory called `/dev`. I'm **NOT** going to go into detail about how Linux finds and creates these files or how you can customize them – that is full-court tech geek voodoo and I'm not going there. You don't need it; you just have to know that `/dev` is where you need to look. I'm a Mac user so I've got a bunch of stuff lying around for use on a Mac which hasn't seen a serial port in over a decade. These days, most laptops and even a few desktop systems no longer have serial ports. However, everything has a USB port and probably the most common USB device to plug in there (other than a hard drive) is a USB/serial converter dongle to run all those serial devices we still have. I own a Keyspan USA-19HS converter that I use for lots of things and since it has one of the standard

USB/serial converter chips at its core, it works fine on a Linux system as well. The reason that I'm using a USB/serial converter is to employ my good ol' Atmel AVR ISP programmer. This is a serial port based device that gets its power from the target board that it is programming.

When I plugged in this USB/serial converter, the Ubuntu system saw it and installed a `/dev` file to reference when talking to it. But where? I did some sleuthing before and after I plugged the device in and found that a directory `/dev/serial` appeared coincidentally after the system saw the USB/serial converter. In `/dev/serial`, two other directories appeared: `by-id/` and `by-path/`. Don't ask me what those are supposed to mean – I don't know – but under those directories were entries for my Keyspan dongle, clear as day. See **Figure 1** for what I found.

Note the highlighted section of the listing: `././ttyUSB0`. This means that the device handle that we want to give to `avrdude` to program our parts is `/dev/ttyUSB0`. Cool, that was easy. Now, let's use that information and a bunch more stuff to set up `avrdude` to program the hex file created when `Eclipse` and `avr-gcc` compiled our source code. Oh, by the way, I got Linux to tell me where that `dev` file was linked to by using the command `"ls -l"` (minus the quotes). Just in case you'd like to have fun with that command ...

Figure 1. Where is my dongle?

```

dcl@ubuntu: ~
File Edit View Terminal Help
dcl@ubuntu:~$ ls -l /dev/serial/*
/dev/serial/by-id:
total 0
lrwxrwxrwx 1 root root 13 2010-02-13 21:27 usb-Keyspan_a_division_of_InnoSys_In
c_Keyspan_USA-19H-if00-port0 -> ././ttyUSB0

/dev/serial/by-path:
total 0
lrwxrwxrwx 1 root root 13 2010-02-13 21:27 pci-0000:00:1d.0-usb-0:1:1.0-port0 ->
././ttyUSB0
dcl@ubuntu:~$ #
dcl@ubuntu:~$

```

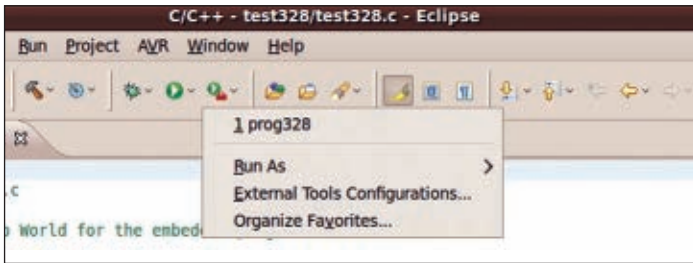


Figure 2. Making an external tool.

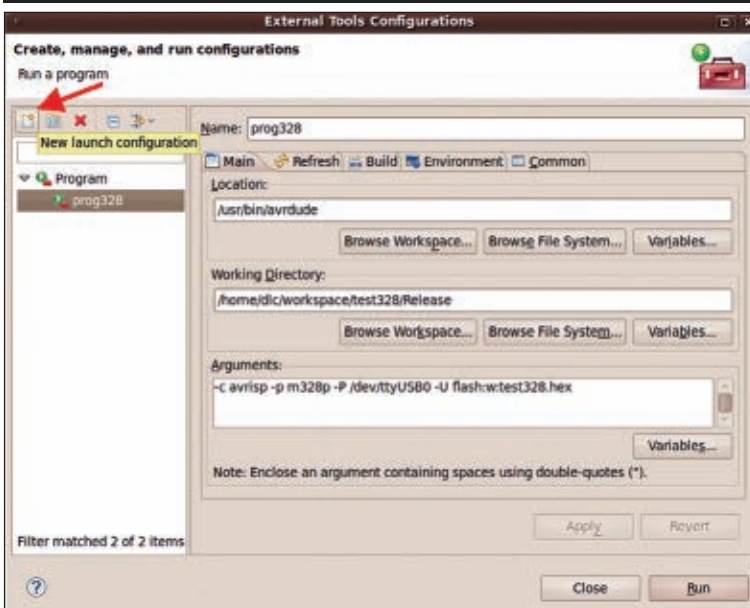
Creating an Eclipse programming tool button.

You can use avrdude on the command line to program your devices, but why? We're already using the Eclipse IDE to write and compile our programs; let's use it to program our boards too! In **Figure 2**, you can see the *External Tools* button and the menu that comes up when you press the *right side of the button*. If you just press the button, you'll execute the default action which will run your program (which it can't do right now). We want to configure a new external tool, so press the right side of that button where the down-arrow-looking thing is. Now select the *External Tools Configurations* menu selection and you'll get a dialog box like what you see in **Figure 3**.

In the upper left side of this window is the *New Launch Configuration* button (see the red arrow in **Figure 3**.) Click it and fill in the blanks as I detail here. Your working directory will not be `/home/dlc/workspace` obviously, so use the location that you selected when you first ran Eclipse. There are four windows that you will have to fill in properly:

Name: This is what you will call your tool to program your project into the AVR part.

Figure 3. Create the programmer tool.



Location: This is where the program you'll run is located. If you followed my directions from last month, then avrdude will be in `/usr/bin/avrdude`.

Working Directory: This is where your `.hex` file will be located. If you stayed with the defaults that Eclipse offered you (and you should, just to keep your life simple), then this will be: `/home/<login>/workspace/test328/Release`.

Arguments: These are the command line arguments that you feed avrdude to program your part. I'll break this line down in a moment; this is the most critical part of the tool setup.

About avrdude arguments. Let me break down what is written here, which deals with all of the hardware as I've described it so far:

-c avrisp. The `-c` flag selects the download hardware. In my case, it is the AVRISP. There are lots of different programmers that you can have. If you look in `/etc/avrdude.conf`, you will see all of the ones that your avrdude install knows about. If you have an AVRISP II (the USB version), then you'll have to compile avrdude yourself with the USB extensions in it so that you can simply say `-c usb`. For some reason, no install ever comes with avrdude configured this way. Go figure. You can locate the procedure here: www.arduino.cc/playground/Bootloader/Ubuntu.

I know ... it talks about the bootloader, but the procedure for building avrdude for use with USB programmers is priceless. Every other place I looked had you jumping through hoops with lex, yacc, bison (huh?). Some of these Linux hacks must just love pain. If you want straightforward AVR information, keep wandering through the Arduino playground. These guys know their stuff and know that you don't want to be a compiler writer — you just want to play with your AVR boards. Good stuff.

-p m328p. This defines the part you are programming. If you type "man avrdude" in a terminal window, you will get the man page for avrdude. It will tell you all the parts that you normally want to use. The `/etc/avrdude.conf` file will have a full listing of the currently supported AVR micros, as well.

-P /dev/ttyUSB0. This is where your serial port linked for access to the USB/serial dongle. It is how you talk to your programmer.

-U flash:w:test328.hex. "man avrdude" gives you all of the possible options for the `-U` flag. This is telling avrdude just what you want to do with your AVR micro. This invocation programs the Flash program space.

After you get all of this entered, you can simply hit the Apply and then the *Close* buttons to save the configuration.

If you have a "vanilla" MEGA328 part that you just bought, its default setup configuration is the

internal 1 MHz oscillator. Recall from last month when I set up our Eclipse system and tested the compiler I set the clock speed to 16 MHz — what I run my stuff at so that I can use Arduino bootloaders. Your brand new part won't have that clock speed; it will be 1 MHz. I suggest that you start another project for a 1 MHz clock if you don't want to dive into avrdude/AVR configuration bits voodoo at this time. If you want a faster setup that has a faster clock, then continue reading.

WARNING: If you make a mistake playing with the AVR FUSE registers, you will "brick" your micro — it won't be broken or anything, but you'll need an STK500 or similar to reprogram the part to get the clocks back. To use the In-circuit System Programming of the AVRISP or AVRISP II, you need to have a working clock. Messing with the oscillator options and getting it wrong will leave you unable to program using the ISP programmers. (You have been warned!)

Now, here is how to set up your FUSE registers on an ATMEGA328. This process will differ in the details for each AVR part; the essentials are the same, but the bits and/or bit locations differ. If you have an ATMEGA328 on a board that you got from someone selling the boards, it may already be configured to work with the clock system on that board. Find out from your supplier what you have before you continue on; you may not need to do this next step.

You will need a terminal window for this step. Type in the following command line **exactly as it is written** to configure your ATMEGA328 to use an external crystal or resonator for high speeds. In my case, I used a 16 MHz resonator:

```
avrdude -c avrisp -p m328p -P /dev/ttyUSB0 -U
efuse:w:0x00:m -U hfuse:w:0xdf:m -U
lfuse:w:0xf7:m
```

A whole lot of stuff is going to be printed in your terminal window as this command does its thing. If everything wrote and verified properly, you're done. If not, check your connections, your typing, and your assumptions; one of them is incorrect.

You're now set for higher clock rates. As soon as this command is complete, you will be in the new mode. I strongly suggest that you have your AVR set up on your protoboard with a resonator or crystal circuit before you run this command. Remember, the above incantation requires that your configuration is exactly as I've been describing in this set of articles.

Running your new programmer tool.

Now that you have everything that you need configured and your IDE setup and your compiler working (remember, you got it working last month), now you are

```
<terminated> prog328 [Program] /usr/bin/avrdude
avrdude: AVR device initialized and ready to accept instructions
Reading | ##### | 100% 0.05s
avrdude: Device signature = 941c58f
avrdude: NOTE: FLASH memory has been specified, an erase cycle will be performed
To disable this feature, specify the -B option.
avrdude: erasing chip
avrdude: reading input file "test328.hex"
avrdude: input file test328.hex auto detected as Intel Hex
avrdude: writing flash (324 bytes):
Writing | ##### | 100% 3.03s
avrdude: 324 bytes of flash written
avrdude: verifying flash memory against test328.hex:
avrdude: load data flash data from input file test328.hex:
avrdude: input file test328.hex auto detected as Intel Hex
avrdude: input file test328.hex contains 324 bytes
avrdude: reading on-chip flash data:
Reading | ##### | 100% 2.94s
avrdude: verifying ...
avrdude: 324 bytes of flash verified
avrdude done. Thank you.
```

Figure 4. Ta da! You have programmed your micro.

ready to program your ATMEGA328 micro. All you have to do is press the tool button on the right side (with the down arrow thing) and select the *prog328* program. You'll program your part with the compiler/linker generated .hex file you told it to use. For subsequent programmings, all you will need to do is press the button — the last tool used will be the default. I have several programmer scripts in my toolbox. I select the one I want based on the project I'm doing at the moment. When you press that button, you should see something like **Figure 4** printed out in your Console window in Eclipse.

Conclusions: What you have learned.

This concludes my mini tutorial about how to install the avr-gcc tools, Eclipse, avrdude, and all that you need to develop and program AVR microcontrollers on Ubuntu Linux. This very same environment can be installed and set up on a Windows or Apple OS X system. The OS X and Linux versions are nearly identical since they are both UNIX variants. I started with a brand new installation of Linux on my laptop and installed everything exactly as I described here. It worked the first time.

If you have an existing Linux system, be very careful in your procedure so that you don't end up with conflicting path names. Make sure that you don't have older installations lying around whose path may come first in your \$PATH list. I recommend that you wipe out any previous avr-gcc installs and start fresh with the latest stuff. It could save you grief along the way and help you avoid pulling your hair out trying to find out why a compile won't work or why avrdude doesn't know how to program a ATMEGA328 micro.

I hope this short tutorial was helpful to those of you that are of that persuasion. I learned a bit myself! My mailbox started to fill up with new and interesting questions that I hope to answer in the near future — enough to keep me busy for a while actually! If you have a question, please don't hesitate to send it on to roboto@servomagazine.com and I'll do my best to help you out. **SV**

EVENTS

Calendar

ROBOTS.NET

Send updates, new listings, corrections, complaints, and suggestions to: steve@ncc.com or FAX 972-404-0269

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: <http://robots.net/rcfaq.html>

— R. Steven Rainwater

MAY

2 **Hawaii Underwater Robot Challenge**
Richardson Pool, Pearl Harbor Naval Station, Honolulu, HI

ROV teams engage in timed, multitasking missions with tethered vehicles.

www.marinetech.org/rov_competition

3-8 **ICRA Robot Challenge**
Anchorage, AK

The challenge includes several events such as Mobile Microbotics, Mobile Manipulation, human-robot interaction, and a virtual manufacturing challenge.

<http://icra.wustl.edu>

7 **SPURT**
Technology Park Warnemunde Ltd., Rostock-Warnemunde, Germany
SPURT stands for School Projects Using Robot Techniques and the goal is to build robots that race each other on the SPURT track.
<http://spurt.uni-rostock.de>

8 **Austrobot**
University of Applied Science, Wels, Austria
Eurobot regional. This year's event is called "Feed the World."
www.austrobot.info

8 **DPRG RoboRama**
Dallas, TX
The first of the twice annual competitions put on by the Dallas Personal Robotics Group. The competition

consists of several events for autonomous robots.
www.dprg.org

8 **RobotFest**
Lawrence Technological University, Southfield, MI
Lots of events including game competitions between two autonomous robots, a robot exhibition, the robofashion show, a Mini-Urban challenge, fire-fighting contest, and a VEX contest.
<http://robofest.net>

13 **NATCAR**
UC Davis Campus, Davis, CA
NATCAR is a high speed line-following contest for large (up to 14" wheelbase), autonomous, car-like robots.
www.ece.ucdavis.edu/natcar

14 **RobotX**
Polytechnic University of Bucharest, Romania
Regional Eurobot event for autonomous robots.
www.robotx.ro

15 **Western Canadian Robot Games**
Calgary Aerospace Museum, Calgary, Alberta, Canada
Events include Sumo, mini-Sumo, line-following, mine sweeper, LEGO Mindstorms, BEAM robots, and an Artbot contest.
www.robotgames.com

27-30 **Eurobot**
Rapperswill, Switzerland
This year's Eurobot championship for the student autonomous robot contest is called "Feed the World" and the robot that collects the most fruits, vegetables, and seeds is the winner.
www.eurobot.org

JUNE

3-5 **ION Autonomous Lawnmower Competition**
Beavercreek, OH
Autonomous lawn mowing robots compete to see which can mow the grass faster.
www.automow.com

4-7 **AUVS International Ground Robotics Competition**

Oakland University, Rochester, MI

Autonomous ground robots must navigate an outdoor obstacle course within a prescribed time while staying within a 5 mph speed limit.

www.igvc.org

5-6 **Motodrone AFO Competition**

Finowfurt, Germany

AFO stands for Autonomous Flying Objects. Robots are judged on their ability to hover in changing wind conditions, fly stably between way points, photograph objects, recover from unexpected dives, and perform precise takeoff and landings.

www.motodrone.de

6-8 **International Micro Air Vehicle Competition**

Braunschweig, Germany

MAV Surveillance (smallest to complete course wins), MAV Endurance, Ornithopter Competition, and Design Competition.

www.motodrone.de

24-26 **MATE ROV Competition**

University of Hawaii, Hilo, HI

Remotely operated underwater robots built by student teams compete to complete a series of tasks.

www.marinetech.org/rov_competition/2010

26-27 **International Autonomous Robot Contest**

San Diego County Fairgrounds, San Diego, CA

Events in this competition include Technical Presentation, Urban Challenge, and the Gold Rush Challenge.

www.iaroc.org

JULY

7-11 **Botball National Tournament**

Edwardsville, IL

Robots move black and white balls on a game board. A robot of appropriate size is provided in kit form.

www.botball.org

11-15 **AAAI Mobile Robot Competition**

Atlanta, GA

Events include Semantic Robot Vision Challenge, Human-Robot Interaction Challenge, Integration Challenge, and Robot Exhibition.

www.aaai.org/Conferences/conferences.php

13-18 **AUVS International Underwater Robotics Competition**

Space and Naval Warfare System Center, San Diego, CA

Autonomous underwater vehicles must complete an underwater course with various requirements. Bots cannot be greater than six feet long x three feet wide x two feet deep, and not be greater than 100 kg mass.

www.auvsi.org/competitions/water.cfm

17-20 **RoboBombeiro**

San Miguel Pavilion, Guarda, Portugal

This is a fire-fighting robot contest.

<http://robobombeiro.ipg.pt>

18-23 **CIG Car Racing Competition**

Barcelona, Spain

This is a competition for race cars controlled by artificially evolved neural nets.

www.wcci2010.org

19-24 **K'NEX K*bot World Championships**

Las Vegas, NV

This event includes Two-wheel drive K*bots (autonomous), Four-wheel drive K*bots (autonomous), Cyber K*bot Division (RC), and Motorless K*bot Division (simple machines).

www.kbotworld.com

"THE PHANTOM DRAW"

The **KILL A WATT** meter is the best way to help you determine your actual energy draw in **ON and OFF** home appliances.

To order call

1 800 783-4624 or online www.servomagazine.com



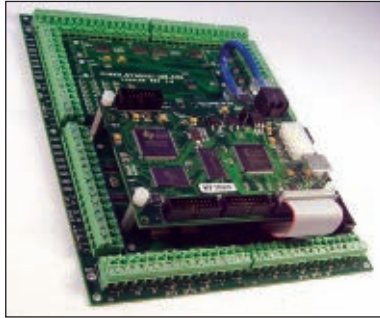
Only \$29.95

NEW PRODUCTS

CONTROLLERS & PROCESSORS

Kanalog 1.0

Kanalog from Dynomotion adds analog and digital I/O to their KFLOP motion controller. Kanalog provides six types of various I/O which is enough to completely drive many types of machine tools.



Standard $\pm 10V$ analog outputs may be used to drive analog motor drives. The analog outputs are true 12-bit DACS — not filtered PWMs that have slow response and ripple. All (eight) analog inputs and (eight) analog outputs are updated every servo sample time of $90 \mu s$ (11 kHz).

Relay drivers, opto inputs, opto outputs, differential encoder inputs, and LVTTTL inputs and outputs are all included along with 112 screw terminals.

The photo shows Kanalog with a KFlop mounted with it and two cables that connect the two boards.

Setting Options

Set options on the KMotion Executive Program for both Kanalog and the required KFlop.

Analog Status

All of the Analog ADC readings and DAC settings can be observed by selecting the Kanalog tab of the analog I/O screen.

Digital Status

All of the digital I/O can be observed by selecting the Kanalog tab of the digital I/O screen.

Inputs are all on the left side of the screen and outputs are on the right. Outputs may be toggled by clicking on the state.

For further information, please contact:

DynoMotion

Website: www.dynomotion.com

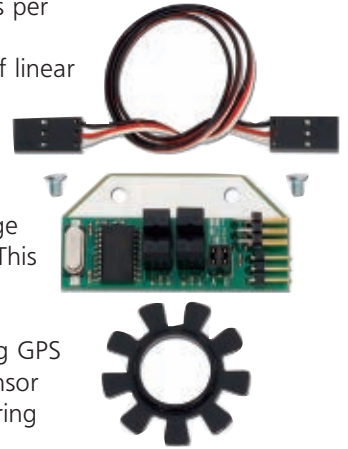
ENCODERS

Position Controller Kit

A new position controller from Parallax uses a quadrature encoder system to reliably track the position and speed of a wheel at all times. With the plastic encoder disk included in the kit, the position controller has

a resolution of 36 positions per rotation; this equates to approximately 0.5 inches of linear travel per position using six-inch tires (not included). The position controller calculates and reports position and average speed data on command. This leaves the main processor free to handle more important tasks like reading GPS coordinates, processing sensor information, and maneuvering complex environments.

The position controller is compatible with any microcontroller via a single-wire half-duplex asynchronous serial communication (UART) bus. Up to four position controller devices can be controlled on the same bus to minimize I/O requirements. For increased functionality, multiple position controllers can be interfaced with HB-25 motor controllers (available from Parallax and sold separately) to control the position of the wheel and automatically provide smooth speed ramping, as well as accurate position advancement capability.



Features:

- Single I/O line can control up to four position controllers.
- 36 encoder positions per revolution.
- Compatible with any microcontroller.
- +5 volt supply for position controllers.

Kit Contents:

- Position Controller
- Plastic Encoder Disk
- 14" Servo Extension Cable
- 4/40 Phillips flat head screw 3/16" (x2)

Retail price is \$29.99. For further information, please contact:

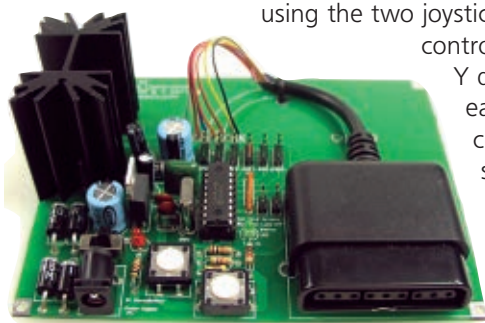
Parallax, Inc.

Website: www.parallax.com

MOTOR CONTROLLERS

Dual Shock Servomotor Controller

Jimages's new DS2 servomotor controller controls six hobby servomotors. The servomotors are controlled



using the two joysticks on the controller. The X and Y direction of each joystick controls a servomotor. This totals four servomotors. To control servomotors numbered 5

and 6, one presses and holds the left shoulder button, while using the right joystick.

Servomotor speed is proportional to the tilt of the joystick. The retail price is \$99.95. Servomotors and PlayStation controller are not included.

For further information, please contact:

**Images Scientific
Instruments, Inc.**

109 Woods of Arden Road
Staten Island, NY 10312
718 • 966 • 3694 Fax: 718 • 966 • 3695
Website: www.imagesco.com

SERVICES

New Custom Front Panel Service

Having helped pioneer the facility for engineers to purchase PCBs online (pcb-pool.com), Beta LAYOUT has announced the introduction of a new online front panel service.



This new service enables users to configure their own front panel designs and place an order directly online. As well as providing a professional "free-to-download" design software, numerous machining options, material thickness, fonts, colors, and finishes are available.

The free, easy-to-use, design software simplifies the configuration and ordering of custom front panels. You can choose from many standardized construction units (e.g., ventilators, sub-connectors) which are available in the software's library. The software even calculates the price of the finished front panels for you. Once your front panel design is complete, an order can be placed directly through the software itself.

Alternatively, DXF files from any CAD program can be

converted and processed for a small surcharge. Separate printing and inscription files are required for this option.

Various front panel materials can be selected from a wide range of natural or colored anodized aluminium and plastic (acrylic). Material thickness from 1.5 mm to 3 mm can be selected. The minimum dimensions possible are 30 mm x 30 mm, up to a maximum of 300 mm x 460 mm. High precision CNC machining such as drilling (with and without threads), countersunk drills, flat milling, and cut-outs are all possible. Ultra-modern milling and drilling machines are used to complete the manufacturing process. The inscription of front panels with text, logos, images, scalings, etc., is achieved using engraving methods and/or high-resolution digital printing. The maximum order quantity is 50 pieces; lead times are available from three to eight working days.

For further information, please contact:

PCB-POOL

Tel: 877 • 390 • 8541
Email: sales@pcb-pool.com
Website: www.panel-pool.com

MISCELLANEOUS

RB-See-66

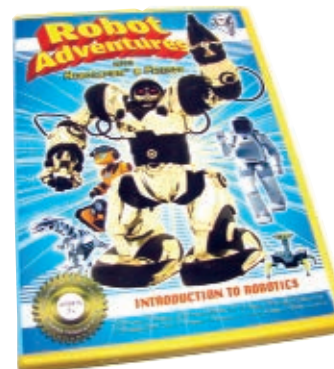
The SeeedStudio DSO nano pocket 1 MHz two-channel digital



storage oscilloscope is a pocket size digital storage oscilloscope based on the ARM Cortex™-M3 compatible 32-bit platform. The oscilloscope is equipped with a 320 x 240 color display, SD card capability, USB connection, and chargeable batteries, and weighs just 60 g. This DSO nano comes with a mini probe, alligator clip, lithium battery, protection bag, and manual. The oscilloscope features automatic measurement of frequency, cycle, duty, Vpp, Vram, Vavg, and DC voltage. It performs precise vertical and horizontal measurements with markers.

RB-Vop-01

Vision One Pictures now presents "Robot Adventures' Introduction to Robotics" which is designed to inspire the next generation of scientists and engineers. This DVD is packed with educational content on robotics, and is hosted by Robosapien and his friends. It helps youngsters understand the fundamentals of robotics. This entertaining and educational DVD is an essential tool for children age 7 and up who are interested in technology. This is the



first in a four-part series.

RB-Boa-05

DMP Electronics has produced a second version of the Roboard — the RB-110. Improvements include changing the CPU to version D of Vortex86DX, a 32-bit x86 CPU (running at 1,000 MHz with 256 MB DRAM) which means no additional IDE driver is needed when installing Windows XP. The board includes an FTDI FT232HL high-speed UART, which supports up to 12M serial speeds. The new board also includes power protection in case the polarity on the power pins is reversed. Just like the RB-100, the RB-110 is compatible with Windows, DOS, and Linux, and open-source code is available. Just like its predecessor, the board measures only 96 x 56 mm.

For further information on these three items, please contact:



RobotShop

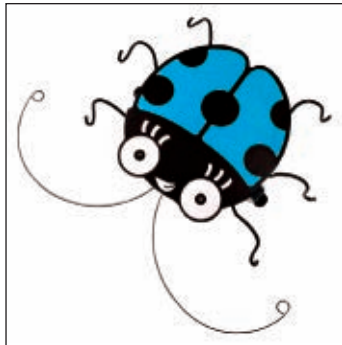
Website: www.robotshop.com

ROBOT KITS

The BeetleBot

The BeetleBot started as Jérôme Demers' high school science fair project. He built an extremely simple and effective circuit using just a pair of cleverly wired switches, two batteries, and two motors. This robot would zoom around an enclosure, cleverly bouncing off all obstacles using its antenna-like bump sensors. Jérôme's goal behind this project was to demonstrate how you can achieve a complex behavior with minimal design and simple parts. True to this aim, the BeetleBot managed to perform its function and purpose without using any electronics or programming. This project earned Jérôme several awards, and a trip to the 2001 International Science Fair.

Jérôme is currently doing an internship at Solarbotics, and has turned his creation into the BeetleBot Solderless Kit. This kit features the same clever design as the original, but assembles with plug-in connectors, and simple screw



and plastic construction. It was important to Solarbotics to make this solderless kit approachable for hobbyists of all skill levels.

The robot is built on a laser-cut acrylic base that is marked and prepared for all screws and component layouts. The pre-assembled wire harnesses plug into the switches, batteries, and motors. The included wire sensors are pre-formed to give the robot a unique insect-like appearance and provide efficient detection. One of the most fun parts of building the kit is adding personality to the robot shell with the included stickers.

The BeetleBot is currently available at a special introductory price of \$34.95 (\$5 off the \$39.99 retail price). You can choose between the happy "Ladybug" or evil "Tribal" designs — each available in four different colors.

For further information, please contact:

Solarbotics, Ltd.

Website: www.solarbotics.com

ACCESSORIES

Cable/Accessory Organizer

Skooba Design now has a new cable/accessory organizer, as well as a new collection of multi-use protective wraps, to its line of tech-travel carrying cases and



accessories. The Cable Stable DLX is a unique accessory storage concept which looks and opens like a book, and features a grid of elastic hold-downs in different sizes and orientations. The elastic strips "float" freely under reinforced retention strips, so they can easily stretch and contract to accommodate a nearly limitless assortment of portable gadgets, cords, adapters, and other accessories.

With the storage grid, pockets on the opposite side, elastic loops for batteries, pens, and thumb drives, and an exterior document/CD pocket, the Cable Stable DLX offers approximately 18 different storage spaces in a 12 ounce case. The new Cable Stable DLX is available at a suggested retail price of \$39.95.

For further information, please contact:

Skooba Design

Website: www.skoobadesign.com

ALL ELECTRONICS

C O R P O R A T I O N

THOUSANDS OF ELECTRONIC
PARTS AND SUPPLIES

VISIT OUR ONLINE STORE AT
www.allelectronics.com

WALL TRANSFORMERS, ALARMS,
FUSES, CABLE TIES, RELAYS, OPTO
ELECTRONICS, KNOBS, VIDEO
ACCESSORIES, SIRENS, SOLDER
ACCESSORIES, MOTORS, DIODES,
HEAT SINKS, CAPACITORS, CHOKES,
TOOLS, FASTENERS, TERMINAL
STRIPS, CRIMP CONNECTORS,
L.E.D.S., DISPLAYS, FANS, BREAD-
BOARDS, RESISTORS, SOLAR CELLS,
BUZZERS, BATTERIES, MAGNETS,
CAMERAS, DC-DC CONVERTERS,
HEADPHONES, LAMPS, PANEL
METERS, SWITCHES, SPEAKERS,
PELTIER DEVICES, and much more....

ORDER TOLL FREE
1-800-826-5432

Ask for our FREE 96 page catalog

iTelegraph



Send Morse code to your
friends using your vintage
telegraph set or built-in
sounder over the Internet!

- * Arduino compatible
- * Open source
- * Hack it for other uses!

For kits, motor controllers,
parts, Arduinos, shields
and more, visit:

www.criticalvelocity.com/itg

Firgelli

www.firgelli.com

LINEAR SERVOS



L12-R Linear Servo

Direct replacement for regular rotary servos
Standard 3 wire connectors
Compatible with most R/C receivers
1-2ms PWM control signal, 6v power
1", 2" & 4" strokes
3 - 10 lbs. force ranges
1/4" - 1" per second speed ranges
Options include Limit Switches and
Position Feedback

L12-NXT Linear Servo

Designed for LEGO Mindstorms NXT[®]
Plugs directly into your NXT Brick
NXT-G Block available for download
Can be used with Technic and PF
Max Speed. 1/2" per sec.
Pushes up to 5lbs.
2" & 4" strokes



PQ12 Linear Actuator

Miniature Linear Motion Devices
6 or 12 Volts, 3/4" Stroke
Up to 5 lbs force.
Integrated position feedback or
Limit switches at end of stroke
External position control avail.

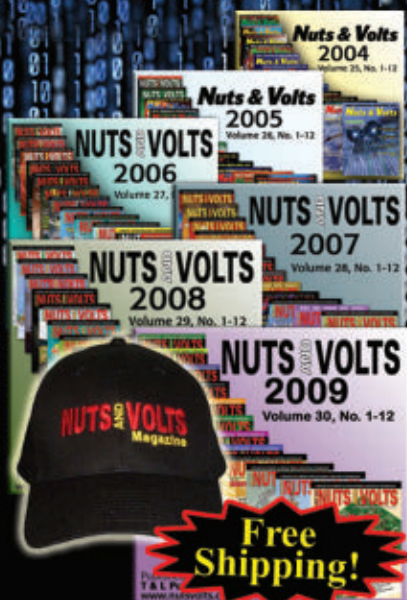


Available Now At
www.firgelli.com

Robotics Showcase

Nuts & Volts 6 CD-ROMs & Hat Special!

That's 72 issues.
Complete with supporting
code and media files.



Free
Shipping!

Only \$129.95

Finally! Full motion control for differential drive robots!



WheelCommander™

- Closed-loop position, angle, velocity & rotation rate
- Real world measurement units
- I2C and RS232 auto-switching interface
- WheelWatcher™ compatible
- Windows API & PID tuning wizard
- Examples for common robot controllers

NU-BOTICS™

www.nubotics.com

WEIRDSTUFF® WAREHOUSE

Software, Computers, Electronics, Equipment, Doo-hickies

384 W. Caribbean Dr.
Sunnyvale, CA 94089

Mon-Sat: 9:30-6:00 Sun: 11:00-5:00

(408)743-5650 Store x324

 **WE BUY AND SELL EXCESS
& OBSOLETE INVENTORIES!**

FREE COMPUTER RECYCLING

We recycle computers, monitors, and
electronic equipment. M-Sat 9:30-4:00



 **WEEKLY SEALED BID SALE AUCTION**
Hi-tech items, electronics
test equipment, and more!

GIANT AS-IS SECTION

10,000 sq. ft. of computers, electronics,
software, doo-hickies, cables, and more!



also check out our...

 **Store**
stores.ebay.com/WeirdStuff-Inc

WWW.WEIRDSTUFF.COM

bots IN BRIEF



UAS DELIVERS THE GOODS

As part of several innovative programs aimed at revolutionizing war zone resupply, the Marine Corps Warfighting Laboratory recently completed the technology demonstration phase of developing unmanned aerial systems for cargo delivery in the austere, forward-deployed environs of Afghanistan.

Immediate Cargo UAS tested the Kaman K-MAX Burro and Boeing Corporation's A160T Hummingbird. Both demonstrations — which took place separately in Utah at the US Army's Dugway Proving Ground — showcased the ability to support tactical ground and aviation resupply efforts of Marine forces in a harsh combat environment. The Warfighting Lab conducts concept-based experimentation integrating operational concepts with current practices and technology to improve the expeditionary warfighting capabilities of Marine Corps. They took on this task in late 2008.

Kaman and Boeing learned they made the cut in August 2009. Demonstration lead Marine Capt. Amanda Mowry and her team chose Dugway, UT as the testing ground. "We tried to replicate the conditions we need the Cargo UAS to fly in," explained Mowry. Aside from its terrain, Dugway's unpredictable weather and high altitude were pluses for the test.

Burro and Hummingbird each faced unforeseen hurdles, making the demonstration more realistic according to lab officials. Mowry and her team altered scenarios more than once due to bad weather. Both aircraft performed a series of required maneuvers during no fewer than four events over three days. They flew distances of 150 nautical miles at altitudes of 7,500 feet. The aircraft carried over 1,000 lbs. sling-load style. They flew at night through hazardous mountain ranges and

hovered above their drop zones. At its conclusion, Cargo UAS showed that an unmanned aerial vehicle can successfully deliver 10,000 pounds of gear to a site within a 24 hour period. This measure was crucial to the demo, noted Mowry. Marine officials estimate that a UAS resupply could replace a convoy of 16 security vehicles and the four resupply trucks they escort, getting about 100 Marines off the road. The hypothetical unit supported in the demonstration was a company of 180 Marines.

FOR THE BIRDS

Instead of using a plastic owl to scare away other pesky birds, why not try Robop? Developed by Scotland resident John Donald, the robotic peregrine falcon can be controlled with a WAP enabled phone. Donald



suggests moving Robop around occasionally so the other birds don't suspect he's not the real deal. Of course, you can always download the 38 page guide that talks all about the \$4,200 security bird. Be sure to read the part that explains why your bird population may increase when the "psychological warfare device" is employed.

GETTING BOMBED WITH RUM

The US Defense Threat Reduction Agency may have been remembering Hitler's famous bunker or those hidden tunnels in the Middle East when they decided that a Robotic Underground Munition would be a handy thing to have around. A RUM hits the dirt, starts drilling, and then blows itself up. The suicide bomberbot should also be able to communicate, avoid obstacles, and have sensors so it can find its way. Wanna get in on this action? Go to www.fbo.gov/index?s=opportunity&mode=form&id=1021a87dd2c2cfce17d34e4258ae0cd&tab=core&_cview=0 if you have the right stuff to make one.

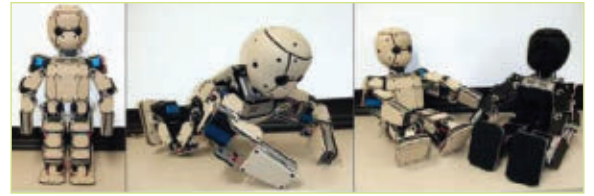
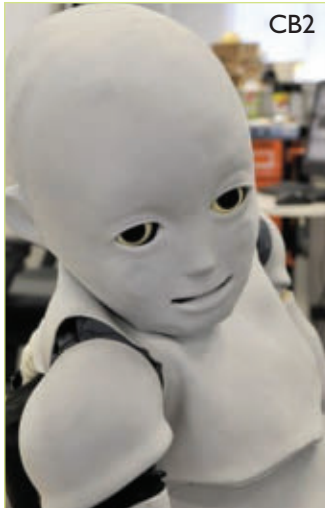


bots IN BRIEF

UGLY BABY

Remember the creepy baby robot CB2? Well, he has a new sibling. Meet M3-Neony. The new baby robot was “born” on March 3rd in Japan to the ERATO project led by Prof. Minoru Asada at Osaka University.

M3-Neony is 50 cm tall and weighs 3.5 kg — just like a real baby which is amazing considering how much tech it holds. It has 22 motors, two CMOS cameras, 90 tactile sensors, and an integrated computer that controls everything. It is fully autonomous and it can crawl, pull itself up holding a table, practice walking, etc. The robot is designed so that even non-robotics researchers can use it as a platform for various research



in human babies' cognitive development and motor learning. Hypothetically, the process of programming M3-Neony to learn to walk will provide insights into how humans figure out the whole process. The M3 stands for “Man Made Man.” Neony is for neonate. There is also an M3-Synchy which is a platform for studying the communication between humans and multiple robots. Both platforms are meant to be available for the research community worldwide.



SAY CHEESE!

Ohio State University (http://artsandsciences.osu.edu/news/rinaldo_robots) has developed an autonomous paparazzi bot that *might* take your picture if you smile nicely. It stands at about the same height as a human and if your expression pleases it, the Ken Rinaldo creation will stop, aim, snap the camera, and upload your image to a social networking site. The bots were invited to the 2010 Olympics, where they undoubtedly had to compete with other paparazzi to get close to Shaun White.

SAY AHHH!

Three Japanese universities (Waseda, Kogakuin, and Showa) and net-tossing security bot robot maker Tmsuk have developed Hanako — a robotic dental patient who behaves just like a human patient (for the most part). The supposedly female robot, can converse with doctors (“Please examine me!” or “That hurts!”), discharge robotic saliva, sneeze, open and close her mouth, etc. The main goal is to offer new dentists or dental students a way to practice “real life” procedures on a robot before working on humans (at Showa University, dental students have to take tests using Hanako). Hanako reacts to mistakes by verbally expressing pain, rolling her eyes, or even simulating a vomiting reflex — thanks to the touch sensors in her mouth.

Standing 157 cm tall, Hanako’s body is the work of Tmsuk while the “medical features” were developed by the dental faculties of the universities involved in the development of the robot. There are nine joints in her body, in places like her jaw, tongue, and even in her eyelids. Now this is an awesome use for a robot!



Cool tidbits and interesting info herein mainly provided by Evan Ackerman at www.botjunkie.com, but also www.robotsnob.com and other places.



LEGO OF MY NIKO

Thanks to the magic of the Internet, you (yes YOU!) can control NIKO N900 — a LEGO MINDSTORMS NXT robot via a hacked Nokia N900 phone.

How does it work?

1. Make or login to a Twitter account.
2. Tweet one of the following commands, preceded by “@N900Niko” — move forward, move backwards, turn right, turn left, or turn around. (In other words, to make the bot move forward tweet “@N900Niko move forward”.)

The Nokia N900 cellphone (associated with NIKO) will receive your tweet within a few seconds and translate it into a message which is then sent to the NXT. The NXT interprets the message sent by the N900, and then translates it into motor movement.

HAPPY HYDRA

Everyone can appreciate a good Hydrazoid. Build your own with a kit from Elenco. It only needs basic tools to assemble and once done, reacts to sound impulses. Clap your hands and it will move forward for 15 seconds. The LEDs and illuminated fiber optic antennas make it more than your average bot on the block. At a size of 7 x 6 x 5", don't forget the two AA batteries (not included).



SERPENTIME

This particular snakebot comes from CMU's Biorobotics Laboratory modular snake robot project.

Snake robots can use their many internal degrees of freedom to thread through very tight areas accessing locations that people and machinery otherwise cannot get into. Moreover, these highly articulated devices can coordinate their internal degrees of freedom to perform a variety of locomotion capabilities that go beyond the capabilities of conventional wheeled and legged robots. The true power of these devices is that they are versatile, achieving behaviors not limited to

crawling, climbing, and swimming. Go to www.cs.cmu.edu/~biorobotics/projects/modsnake/ for more factoids.

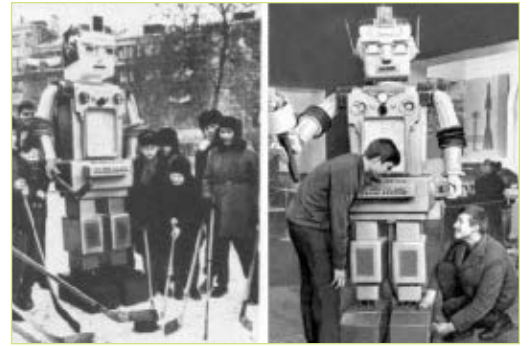
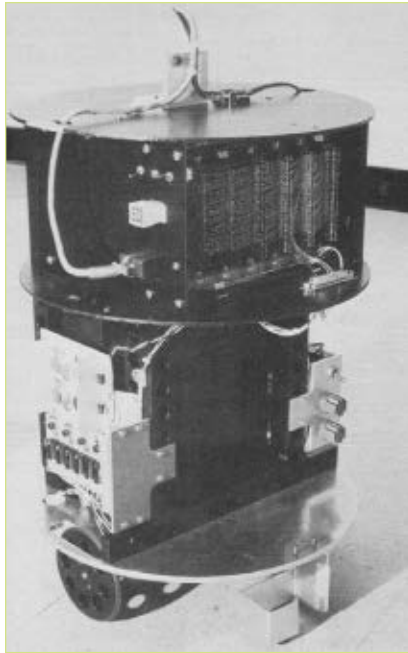
WHAT THE (iPOD) DOCK!?

The Robovie mR2 is a 30 cm tall humanoid robot developed at Japan's ATR Robotics and Communication Laboratories. It sports 18 joints (three in each eye, three in the head, four in each arm, one in the back), 18 servo motors, a CCD sensor 3.4MP camera, two microphones, and a mono speaker (2W). mR2's real trick is its communication channel (between you and him) which is established through an iPod Touch that you need to place inside its chest. You can control this little companion by touching the screen of the iPod Touch, via Wi-Fi (802.11b/g), or Bluetooth.

According to ATR, mR2 can communicate info from your iPod Touch to you by his gestures — some of which could be considered, well, umm, obscene.

This may be why there is no info on pricing or availability since it is not yet commercialized. However, ATR states they are considering that.





BACK TO THE FUTURE

Cybernetic Zoo (<http://cyberneticzoo.com/>) is a website with a whole bunch of incredible pictures and information on early robots from all over the world. You can easily spend hours (or days?) paging through all of the bots. There are also cool commentaries and articles from back in the day that go along with the photos. For example, here's a September 1934 article from the Winnipeg Free Press:

Robot At Toronto Show Turns on Custodian and Smashed Him on Shoulder

Toronto, Sept. 6. Michael Harley, of London, England, custodian of "Alpha," the robot woman being displayed at the Canadian National exhibition, Wednesday crept from the hospital with a sore shoulder and a bad case of the jitters. He went back to look after "Alpha" with some trepidation after she belted him Tuesday with a mailed fist and knocked him to the floor. He was standing with his back to her. He had uttered no word of command, but suddenly the mechanical woman swung on him. Alpha, after turning on her master, stood motionless and expressionless till someone else started ordering her around.

(Maybe this is where all the killer robot panic originally comes from ...)



COMBAT ZONE

Featured This Month:

Features

- 26 BUILD REPORT:**
Reversing the Trend
by Pete Smith
- 27 MANUFACTURING:**
DIY CNC for CAD/CAM
by J. Miles
- 30 PARTS IS PARTS:**
Kitbots Rolls Out B-16 Gearmotor Mounts
by Kevin Berry
- 31 RioBotz Combot Tutorial Summarized: LaunchBots Summarized** by Kevin Berry
- 34 Combat Zone's Greatest Hits**
by Kevin Berry

Events

- 35 Feb/Mar Results and May/Jun Upcoming Events**
- 35 EVENT REPORT:**
Motorama 2010
by Chris Olin

BUILD REPORT:

Reversing the Trend

● by Pete Smith

I recently completed our first drum bot — Weta, God of Ugly Things (**Figure 1**). And while it is a pretty conventional design, it does have one feature rarely seen in bots of this type: The drum is reversible, so it can spin in either direction.

Drum bots work best when the drum is spinning so that the teeth will come up and catch an opponent and throw them up into the air. If the bot gets flipped over, the teeth then try to push the opponent down instead. This has proven to be pretty ineffective (**Figure 2**).

Weta was designed to be completely invertible and it can operate equally well either way it ends up since it is able to quickly and smoothly reverse the direction of rotation of the

brushless motor that powers the drum. Reversible Electronic Speed Controllers (ESC) have been used for some time in RC cars, but have been cost-prohibitive. The

FIGURE 1

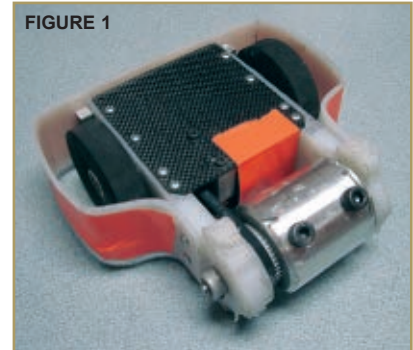
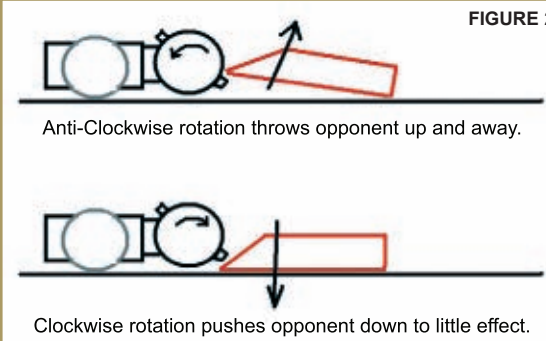


FIGURE 2



recent availability of much cheaper units have made them attractive for use in the smaller 1-12 lb weight classes.

The ESC I used in our beetleweight was the Turnigy TR35A-V2 from www.hobbyking.com and it cost less than \$40. As delivered (**Figure 3**), it has a small fitted fan, long cables, and a separate on/off switch; it weighs a total of 2.5 oz. The fan is too fragile for use in a combat robot; the switch is another failure point; and the wires are too bulky and heavy. I removed the fan (it's held on by only two small screws) and clipped off its power cables close to the ESC's circuit board. The wires for the switch were cut about an inch from the board, stripped, soldered together, then protected by some heatshrink.

The other wires were shortened and new connectors added to match the type I use in my other beetleweights (so we can use the same batteries). There was also a large capacitor that was held in place by the fan, and that was secured using a dab of Shoe Goo®. The result (**Figure 4**) is much neater and almost one ounce lighter.

The ESC is best programmed using the optional programming card (**Figure 5**). The various options are explained in the manual that comes with the ESC. I initially tried out the brake function but found it stopped the drum very abruptly

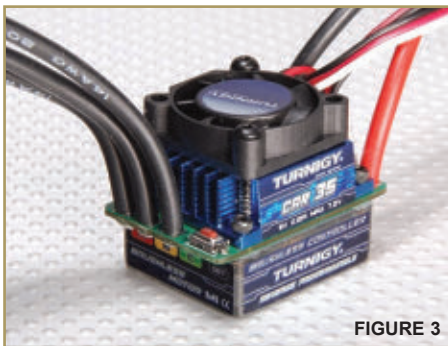


FIGURE 3

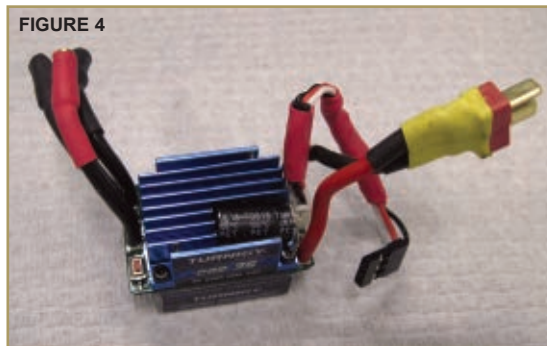


FIGURE 4

and applied too much stress to the pulley on the weapon motor. I reduced the braking settings to zero and found that the drum still stops and reverses quickly enough for it not to be a problem. I used the rudder function on my TX to operate the forward, stop, and reverse of the motor. After a few problems with calibration (the lights did not come on as per the instructions), I had it running well. The rudder stick is spring loaded so it always returns to the center "off" position if released — a useful safety feature. The ESC also failsafes correctly if it loses the signal from the TX.

Weta had its first competition at Motorama in February and the reversibility of the drum really proved very useful and allowed us to win at least two fights we would have otherwise lost. Removing the fan did not create overheating problems since the "cruising" amps to drive the drum were well below the rating of the ESC.

We went on to take 2nd place in the 3 lb class. Weta's

fights can be seen on YouTube by searching for "Weta" and "Motorama."

Hobby King also has 60A and 18A versions which could work in hobby and antweights. The 35A we used and the 60A version both run on 3S LiPo (with the 18A one only for 2S). Some reports say that the 60A will run on 4S, but that has not been confirmed. **SV**



FIGURE 5

MANUFACTURING: DIY CNC for CAD/CAM

● by J. Miles

When I get ready to start building a new fighting robot,

the first thing that I do (besides daydream at work about what I

want it to look like!) is sit myself down in front of a computer and

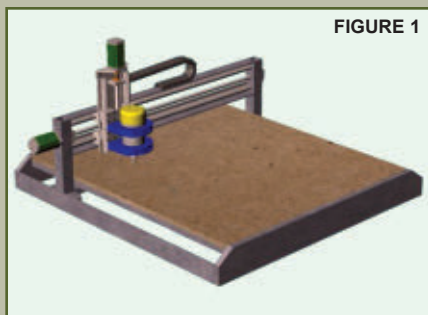


FIGURE 1

spend several hours in my CAD (Computer Aided Drafting) program, drawing my ideas out in a virtual world. I like to design things this way because it's MUCH easier to just hit the undo button when I make a mistake on the computer.

After I have my new, super-slick killer robot all drawn out, I always seem to run into a little problem: how to transform my machine made from pixels into a real live fighting monster. Since I couldn't find a "replicator," I decided to do the next best thing and build a CNC (Computer Numerical Controlled) router to take my CAD design to CAM (Computer Aided Manufacturing).

For those unfamiliar with CNC routers, I am building a three axis machine, meaning it can move in three different directions: backwards and forwards (the Y axis); side to side (the X axis); and up and down (the Z axis). Each direction is moved by a stepper motor that is being told what to do by a computer. A stepper motor is a special type of motor that only turns a small increment (called a step ironically) every time power is applied to it. The computer controls and counts



FIGURE 2

these steps, and that is how it knows where it is while it's cutting something out.

I first contemplated buying plans or a kit of some kind because I knew nothing about CNC routers. I looked around at what was out there, and I decided I wanted a machine made from steel and aluminum. Machines in the size I wanted were out of my price range, so I came up with my own! **Figure 1** shows the CAD drawing of the CNC router I built. It has a usable cutting area of 33" x 33" x 6".

The smallest axis on the machine is the Z since all it has to do is raise and lower the router. To build this axis, I started by using a jigsaw to cut a square and a rectangle out of an aluminum plate (see **Figure 2**).

In **Figure 3**, you can see my completed Z axis made from the two plates that I cut out. To convert the rotary motion from the stepper motor into linear motion to raise and lower the router, I am using a ballscrew/ballnut combo. A ballscrew/ballnut is a lot like a regular threaded rod with a nut on it, but there are ball bearings in between the surfaces that rub



FIGURE 3

together which creates a much smoother motion. When you spin the ballscrew, the ballnut travels up and down depending on which way the screw is spinning — which is how you get the X, Y, and Z axes to move.

A trip to the local steel yard, and I had all my steel to build the machine. I had to have them cut the 20' chunks of steel down to what I like to call "Honda Accord with the back seats laid down" size pieces. That turns out to be about nine feet long. They even threw in the rust for free (see **Figure 4**)!

I decided to make the gantry for my machine (where the X axis sits in) from two inch wide by quarter inch thick angle iron. I used a Sawzall® to cut through the steel and a disc sander to square the ends up. In **Figure 5**, I've started to bolt down the linear bearings for the X axis. I used a center punch first to mark my holes, and then used my drill press to drill them out. There are a lot of different kinds of linear bearings out there; I used some that are made from round rods supported on rails with bearing blocks that slide across them.



FIGURE 4

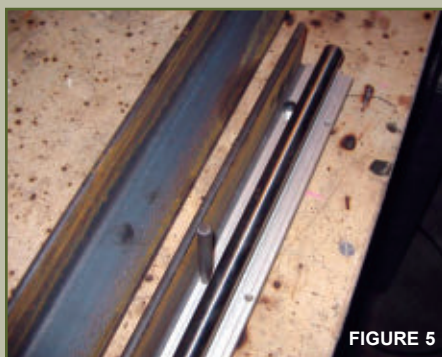


FIGURE 5

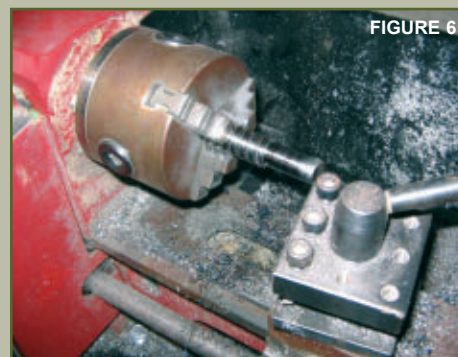


FIGURE 6



FIGURE 7

The ballscrews that I bought came directly from a factory in China. The threads ran right to the end on both sides, so I turned them down on my lathe to fit into the half inch bearings that support each side (see **Figure 6**).

In **Figure 7**, you can see the steel frame, linear bearing rails, and ballscrew laid out in their places to form the gantry. After I welded the steel frame together, I bolted the Z axis into place and slid it side to side to make sure it moved freely.

When I started this project, I didn't have a workbench big enough for the completed router, so I built an extension onto my current one for it to sit on. After I had a big enough area, I cut the steel for the base of the router and laid it into place. When I had everything squared up, I welded everything together. **Figure 8** shows the newly assembled base with the gantry sitting in it. (The soda can on the front corner is for refreshment and scale.)

Figure 9 shows how I connected the ballnut to the axis that it moves in. I used a piece of aluminum angle with one side cut



FIGURE 8

round to match the ballnut.

Mounting the router to the Z axis proved to be a little tricky, since it didn't come with any mounting holes. After some more day dreaming at work, I settled on a two piece clamping mount made from blue plastic. These were made on my cousin's CNC router, and seeing his machine in action made me want to finish mine that much more (see **Figure 10**)!

The last axis to finish was the Y. In **Figure 11**, I've laid out the ballscrew and stepper motor that will move it. I used a flexible coupling to connect the motor to the ballscrew so that even if things didn't line up perfectly, the motor wouldn't bind. Any binding on a machine like this is bad news since the stepper motor can stall and the computer can lose track of where it is (thus ruining the part you are cutting out).

After the Y axis was finished, I started building the cutting table which is what I clamp the raw material to that I will cut parts from. I welded four 1" square tubing bars across the whole machine using a level to keep them even with each other (see **Figure 12**).

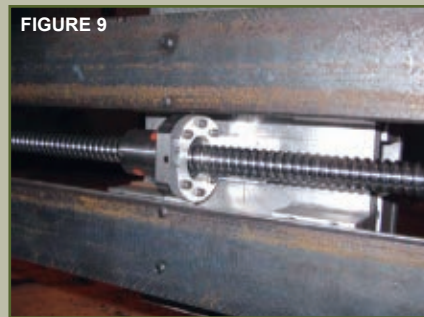


FIGURE 9

You can see all four of the tubes welded into place in **Figure 13**.

After the welds cooled, I clamped a piece of 3/4" thick particle board down that will be the cutting table. I used particle board because it is cheap. Plus, if my router goes a little too deep when cutting something out, I can just fill in the spot with wood putty and sand it smooth to have a flat work surface again. In **Figure 14**, you can see that I also wired up the stepper motors to the computer. At this point, I wanted to make sure everything worked before I painted it in case I needed to make some changes.

Everything worked great, so I took the whole machine apart to paint it. (I still remember the look on my wife's face when she came out into my garage to find me tearing the machine apart shortly after I showed it to her working!) I wanted to be proactive in preventing rust on the steel parts of the machine, so I painted it in the prettiest blue that Walmart's paint department had to offer. (see **Figure 15**).

Figure 16 shows everything put back together; I really like how the machine came out! The black-snake-looking thingies that the wires are



FIGURE 10

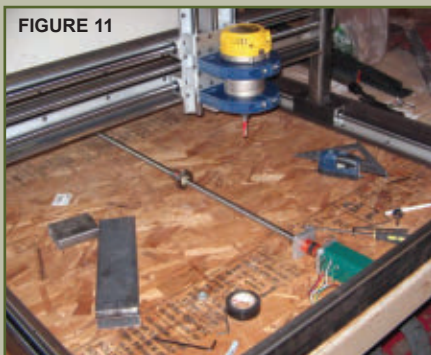


FIGURE 11

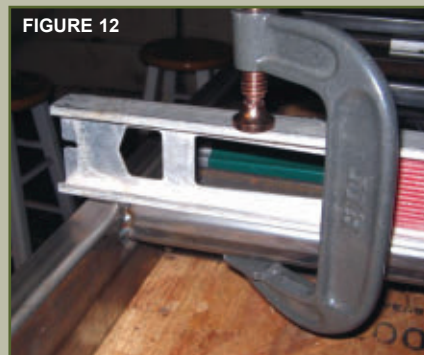


FIGURE 12



FIGURE 13



FIGURE 14



FIGURE 15

going through are called cable tracks. The last thing I want to happen is have the wires on my machine get pinched and shorted out as the three axes move. Cable track keeps the wires neat and tidy. To the left of the machine, you can see the computer and electronics box that runs the show.

I purchased the guts to this box in kit form from www.probotix.com because electronics are not my strong point, and I am very

happy with it. It came with everything I needed to wire up the machine, and after a few phone calls to their tech department (since I don't know how to read an owner's manual), the machine was moving on its own in no time! I built a clear box out of Lexan plastic to hold all the electronics in (see **Figure 17**).

Figure 18 shows the machine doing what it does best — making parts designed in a CAD program.

I'd like to say the first thing I made was a totally awesome killer robot part, but since the wife was so patient with me while I was building it CNC, I figured I would make her a giant "S" to put in our baby girl's room.

I'm really looking forward to seeing my creations on the computer screen come to life, as well as testing the limits of my machine by seeing what kinds of materials it will cut! **SV**

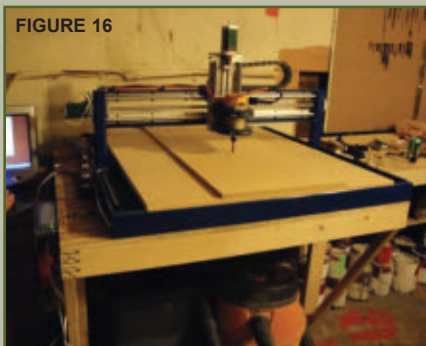


FIGURE 16



FIGURE 17

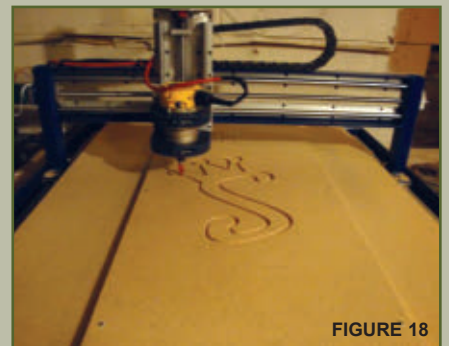


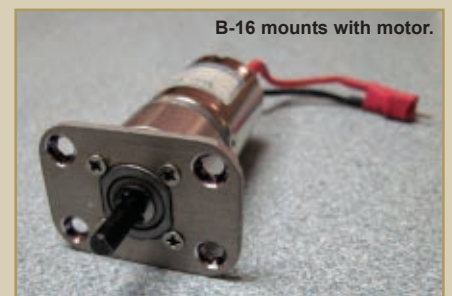
FIGURE 18

PARTS IS PARTS: Kitbots Rolls Out B-16 Gearmotor Mounts

● by Kevin Berry

I have a pretty solid policy that this column only promotes parts that are thoroughly "combat tested" by the fighting community. This month, I'm bending that policy, so to speak, by featuring a new product that is so simple and so

nifty, it's just got to be shared. Team Rolling Thunder — producer of the Kitbots line of products — has come up with motor mounts for the popular Robot Marketplace B-16 beetle gearmotors (www.robotmarketplace.com).



B-16 mounts with motor.

Rolling Thunder has a solid rep in the community for service and quality, and Pete did fight Weta, God of Ugly Things at Moto (see companion article, "Reversing The Trend"), so I feel pretty comfortable endorsing these mounts without the usual body of combat data to back them up. I welcome feedback from folks who use these. **SV**



RioBotz Combot Tutorial Summarized – LaunchBots

● Original Text by Professor Marco Antonio Meggiolaro, Summarized by Kevin Berry

Professor Marco Antonio Meggiolaro, of the Pontifical Catholic University of Rio de Janeiro, Brazil, has translated his popular book – the *RioBotz Combot Tutorial* – into English. Last fall, *SERVO* summarized Chapter 3, "Materials," focusing on commonly used materials for combat bot building. This month, we attempt to do justice to a portion of his exhaustive chapter on weapon design. Chapter 6 – "Weapon Design" – is a college level textbook on the design and operational theory of today's combat weapon systems. In this article, we present a much simplified version of the "LaunchBots" section of this chapter. Marco's book is available free for download at www.riobotz.com.br/en/tutorial.html. For hard copy purchase (at no profit to Marco) go to Amazon, published by CreateSpace. All information here is provided courtesy of Professor Meggiolaro and RioBotz.

Launcher Design

Launchers need to deliver a huge amount of energy during a very brief time. Because of that, they're almost invariably powered by high pressure pneumatic systems. A 4" bore cylinder with 8" stroke

pressurized at 1,000 psi would accelerate a 220 lb mass with an average power of about 566 HP!

Of course, this power is only delivered during a very short time. However, a light-weight electric motor or internal combustion engine cannot supply this unless, of course, the motor is used to store kinetic energy in a flywheel during a few seconds with an ingenious and very strong mechanism that suddenly transfers this energy to the launcher arm – as done by Team Whyachi's Warrior SKF robot (**Figure 1**). Such a sturdy mechanism is not simple to build.

Hydraulic systems are not good options either for launchers. They can deliver huge forces and accelerations, but their top speed is relatively low.

Most launchers try to either maximize the height or the range of the throw. "Height launchers" try to launch the opponent as high as possible and try to flip it while causing damage when it hits the ground. "Range launchers" try to launch

FIGURE 1

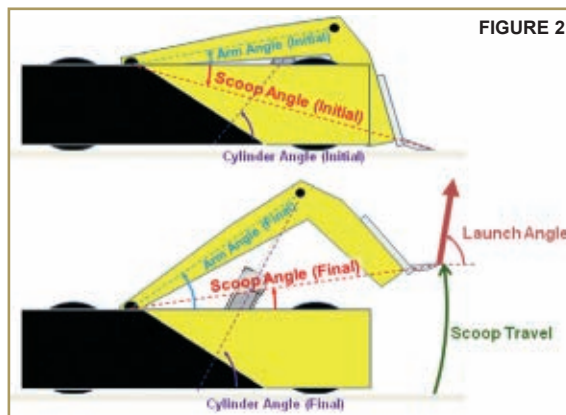


the opponent as far as possible – not necessarily high – trying to throw it out of bounds to the arena dead zone.

Three-Bar Mechanisms

A very popular launcher design uses a three-bar mechanism. The "three bars" are the pneumatic cylinder, the main structure of the

FIGURE 2



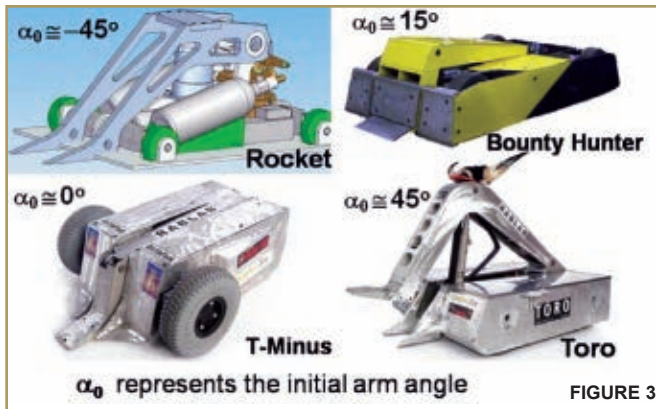


FIGURE 3

launcher arm, and the part of the robot chassis that connects the arm and cylinder pivots. The launcher arm tip features a wedge-like scoop. The initial and final angles of the arm and scoop tip are shown in **Figure 2**.

During the launch, the arm follows a circular path, with an arc length that is usually between one and two times the height of the launcher chassis. The launch angle is approximately perpendicular to the average between the initial and final scoop angles.

Figure 3 shows four different launcher configurations. The lightweight Rocket has both initial arm and scoop angles that are around minus 45°, making it a good “range launcher” due to the average 45° force it delivers. The only problem with this design is that it requires a high chassis to get a sufficiently long arm with that minus 45° angle, decreasing its stability and making it more vulnerable to spinners.

Height launchers Bounty Hunter and T-Minus were able to lower their height when the arms are retracted,

angle of about 15° and 0°, respectively. Their average scoop angle during the launch is close to zero (horizontal), leading to an almost vertical launch angle and force that allows them to throw the opponents very high in the air. Their low initial cylinder angle (below 45°) puts a lot of stress on the back pivot joint of the arm, initially trying to push the arm forward with almost the same force used to launch the opponent. This forward force — which tries to rip off the back pivot — is not necessarily wasted because it doesn’t produce extra work. However, this added force increases the friction losses in the joints. This is the price to pay for a low profile launcher.

Toro, on the other hand, has an initial cylinder angle close to 90°, which doesn’t stress the arm pivot too much. It is also a height launcher because its average cylinder angle is close to zero. To be able to accommodate its relatively long cylinders, it needs to increase its initial arm angle to about 45°, making its tall launcher arm vulnerable to horizontal spinners.

Note the curved strap under Toro’s arm that limits the stroke of the cylinders, avoiding their self-destruction when reaching their maximum stroke.

To properly simulate the



FIGURE 4

launch, it is likely that you’ll need some dynamic simulation software. Or, you can use the spreadsheet from www.hassocks-hog.co.uk/flipper_calculator.htm which has nice launcher models

(as pointed out in the March ‘09 edition of Combat Zone).

As far as scoop design, a long scoop at the end of the launcher arm (as seen in the middleweight Sub Zero, **Figure 4**) makes sure that contact between the robots won’t be lost during the entire stroke of the launcher arm. Be careful though — a very long scoop will be vulnerable to drumbots and undercutters which may bend it until it loses functionality (like not being able to get under robots with low ground clearance).

For range launchers, the ideal launch angle of the arm to maximize the range for very low profile opponents is 45°. If you can’t get under such very low profile opponents and your arm has a very short scoop, then the best launch angle to maximize the reach would be 30° (with respect to the horizontal). (The preceding paragraph grossly condenses a very technical series of discussions and calculations. Combat Zone STRONGLY recommends reading the full text in the tutorial before deciding on arm and scoop angles.)

Since most combat robots tend to have a low profile — to keep their center of gravity low — we can draw several conclusions about range launchers. If the launcher can only provide a small scoop compared to the arm length, then it is better to set its arm such that its launch angle is about 36° from the horizontal, to reach a maximized range. If the scoop is about 25% of the arm length, then

FIGURE 5

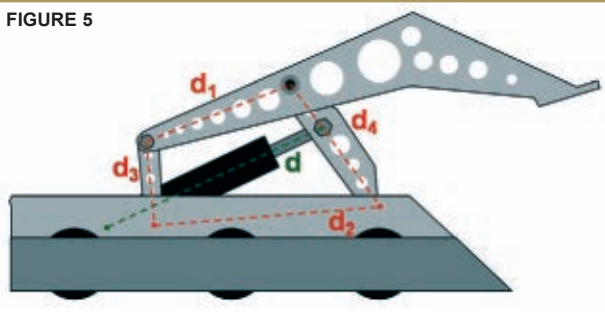
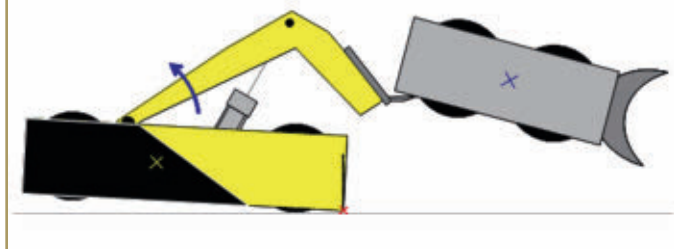


FIGURE 6



a steeper 41° launch angle would be a better option. Finally, if the launcher has a wedge to get under the opponent, or if you decide to use a very long scoop in its arm, then the best choice would be a 45° launch angle. With these values, you spend most of the launch energy throwing the opponent instead of making it spin. Ideally, you should try to keep the opponent's spin as low as possible to maximize the launch range.

Four-Bar Mechanisms

The problem with most range launchers with three-bar mechanisms is that they usually end up with a tall chassis if they want to throw opponents at the ideal launch angles from 36° to 45° .

One alternative is to use four-bar mechanisms. The four bars (as seen in **Figure 5**) consist of part of the launch arm (d1); part of the chassis (d2); and two auxiliary links (d3 and d4). Technically, these launchers have a five-bar mechanism because the pneumatic cylinder (d) counts as a fifth link.

Four-bar mechanisms have two advantages. First, if well designed, they can be completely retracted inside the robot, allowing the use of a low profile chassis. If the constant lengths d1, d2, d3, and d4 are appropriately defined, it is possible to generate optimal trajectories for the arm tip. You can, for instance, make the arm tip trajectory become almost a straight line, with a desired optimal angle (which for range launchers would probably be between 36° to 45° with respect to the horizontal). The four-bar

mechanism calculations are too lengthy to be shown here, but you can make them using a free static simulation program that can be found in the tutorials on Team Insanity's website at www.totalinsanity.net/tut/mechanical/4barfrontbar.php.

Launcher Stability

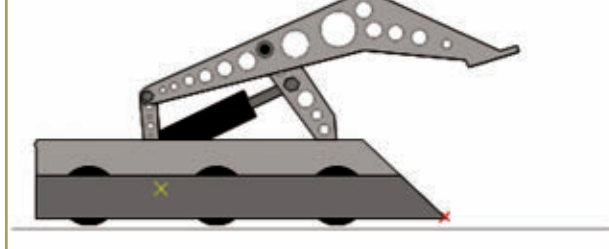
During the launch, the arm should not tilt forward too much or be pushed backwards. Otherwise, it will lose its effectiveness.

Due to the very high forces involved during the launch, it is very likely that height launchers will tilt forward until they touch the ground at their foremost point (shown in **Figure 6**). To avoid tilting forward even more and becoming unstable, the launcher's center of gravity should be as far back as possible.

Another concern with height launchers is with the stiffness and damping properties of their front wheels. During the launch, these wheels are really compressed against the ground, storing a great deal of elastic energy. Towards the end of the launch — when contact with the opponent is almost gone — these wheels will spring back. If their damping is low — like with foam-filled rubber wheels — they may launch the launcher and even make it flip backwards.

Range launchers may tilt forward as well, but it is very unlikely they will lose stability in this way. This is because the line that contains the launch force vector usually meets the ground within the launcher footprint (or very close to its foremost point) due

FIGURE 7



to the shallower launch angles. Range launchers have a problem with very shallow launch angles because the horizontal component of the force may become too large for the wheel friction to bear, pushing it backwards. Tire friction is very important to prevent range launchers from being pushed backwards; decreasing the contact time with the opponent and the effectiveness of the launch. A typical high traction tire is usually enough for a 45° launch angle. For the 36° or 41° angles — which maximize the launch range for the small scoop to arm values — you might need a higher friction tire. If this is not achievable, then the best option is to adopt a higher launch angle.

It is also a good idea for range launchers to locate their center of gravity as far back as possible (see **Figure 7**) to prevent them from even touching their foremost point on the ground; this point will probably have a coefficient of friction with the ground lower than the tires. If some tilting is inevitable, then it might be a good idea to install some anti-sliding material on the bottom of the robot such as a rubber strip to increase friction.

Summary

As you can see, height launchers can benefit from a long scoop, while range launchers should choose average impulse/launch angles between 36° and 45° — depending on the opponent's aspect ratio — as long as they have enough tire friction to not be pushed backwards. **SV**

COMBAT ZONE'S GREATEST HITS

● by Kevin Berry

Last month's debut of this new feature generated a lot of buzz — or should we say crash! I've chosen this month's selection of gourmet damage based on the order I received it (until I ran out of space).

Tim Bouwens of the



DBA.



EPZ.



EPZ After.



Warpig.

Netherlands, builder of VDF of Team Think Tank and Pookie of Team Wazio, describes this fight: It was the third round fight of the 18th AWS (Antweight World Series), with multibot EPZ (Eky Pekang & Zupeng) vs. DBA. At the time, DBA was one of the best antweight spinners in Europe and the fight against the multibot was one of the most destructive and hilarious. The fight started with a couple of big hits, sending the individual multibot against the roof of the arena. After the Lithium battery of Z was ejected out of the robot after another big hit, the fight was stopped and the battery removed. EP then faced DBA alone in a funny cat and mouse game around the remains of Z. Finally, DBA won by knocking EP out of the arena (a way to win in the European competition). The bots were built according to the AWS rules at www.antweight.co.uk/rules.htm.

Troy Mock, from Team Asian Invasion, submitted this report:

Here's damage that happened at Sacktown 6. My antweight, Warpig, fought Gyrobot (a walking robot with a 3/4 pound steel blade). This was at Smacktown in Sactown 6, February 28, 2010. Gyrobot hit the bottom of my 0.09 aluminum wedge, took a large chunk off the

top of the wedge, and hit my aluminum servo cover. Gyrobot's massive blade ripped the whole back off the servo and hit my robot's arm. It chewed up the arm pretty bad, shearing a couple of bolts and bending it. Surprisingly, the servo still worked! I won the match and battled to third place out of 17 robots.

Eric Mueller, in response to a plea for Big Bot Hits, sent this in. "Here's what Super Megabyte did to my LW HazMat in the SHW rumble at the NPC Open. Oddly enough, nothing but the "mondo" power switch broke. I took the front plate off for inspection, but wasn't a result of the hit."

Billy Moon — in the true spirit of a "Greatest Hit" — shows what it's like to be on the receiving end of a hit from the legendary Ziggy. This is his bot Starhawk getting some serious air time.

Before and after photos, brief descriptions of the fight, and builder's name can be submitted to me at LegendaryRobotics@gmail.com. Or, if you have an action shot that clearly shows what's going on, those are welcome too! These don't have to be current. You can submit anything clear back to the good old days, when bots were made out of wood. Anything legal is fair game. **SV**



HazMat.



Starhawk airborne.

EVENTS

Completed and Upcoming Events

Completed Events: February 11 - March 7

Central Illinois Bot Brawl 2010 was presented by the Central Illinois Robotics Club in Peoria, IL on 3/6/2010; 24 bots registered.



Motorama 2010 was presented by the North East Robotics Club in Harrisburg, PA on 2/19 and 2/21/2010; 109 robots registered.



Roaming Robots presented Barnsley Bash 2010 in Barnsley, England on 2/13 and 2/14/2010.



Upcoming Events for May - June

HORD Spring 2010 will



be held on May 15th in Brecksville, OH, presented by the Ohio Robot Club. Go to www.ohiorobotclub.com for more information.

Maker Faire Bot Gauntlet will be held in San Mateo, CA on May 22nd and 23rd, presented by California Insect Bots. Go to <http://calbugs.com> for more information.

Roaming Robots will present shows at Aylesbury on 5/9/2010, and at Guildford on 6/13/2010. Go to www.roamingrobots.co.uk for more information. **SV**

EVENT REPORT: M torama 2010

● by Chris Olin

February 19-21, 84 killer robots gathered in Harrisburg, PA to delight hundreds of motor sports fans with their metal crushing mayhem in the North East Robotic Club's (NERC) Robot Conflict 2010. Robot Conflict is part of the massive Motorama indoor motor sports spectacular held annually at the Pennsylvania Farm Show Complex.

First built in 1931 as an agricultural exhibition hall, the Farm Show Complex has expanded into a 600,000 square foot indoor state fair grounds and expo center. In 1977, a group of motocross riders came up with an idea to use this massive complex for a winter indoor motor sports event. Since that first

event in February 1978, Motorama has expanded to include motorcycle, ATV, Go Kart, Quarter Midget, and RC car racing. The event also features car shows, a beauty pageant, and, of course, combat robotics.

NERC was founded in 2000 by a group of east coast robot builders to provide safe, easily accessible locations in the East for robotic combat events. At that time, all major robotic combat events were held in the western US and the UK. In 2002, NERC was approached by Motorama organizers looking for new attractions for their event. Since 2003, Robot Conflict has been a popular

attraction at Motorama, filling the stands of the livestock show hall with robotic combat fans. Robot Conflict has grown to become the largest robotic combat event east of the Mississippi, and attracts builders from all across the US and Canada.



Main arena floor before combat.



Robot Conflict 2010 started out with fairy weights and ant weights fighting in the 8' x 8' arena on Friday. Six fairies fought a full round-robin tournament, resulting in the well-named vertical spinner Fright Blade finishing 4-2 and taking third place. Mango Farmer, a four wheeled pushing bot, and Crim, another nasty vertical spinner, both finished 5-1. Mango Farmer's only loss was to Crim, therefore Mango was given second place and Crim was declared champion.

Meanwhile, 16 ant weights battled their way through a double-elimination bracket. A robot with an overhead bar spinner weapon and an odd V shaped body named Green Monster fought his way to the winner's bracket semi final where he was sent to the loser's bracket by Maelstrom, a rugged drum spinner with an eerie blue light. Green Monster then faced the faster and well driven four-wheeled pushy bot named Gilbert. Gilbert defeated Green Monster and ascended to the finals to face Maelstrom, while Green Monster settled for third place. The

Battle damaged arena floor.



ant weight final was a high speed, hard hitting dual that ended in a judge's decision giving Maelstrom the championship and sending Gilbert home with second place.

Saturday and Sunday featured four classes of robots fighting double elimination tournaments in the 16' x 16' arena in front of a crowd that varied from a few dozen to over two hundred, depending on the time of day. Action started with the first round of 19 beetle weights. In early rounds, One Fierce Upper Cut, an infamous vertical disk spinner, tore a path of destruction through the brackets until he hit Another Brick in the Wall — a well armored, four-wheeled pushy bot inspired by the Pink Floyd song of the same name. Early in the match, Upper Cut lost main power sending him to the loser's bracket and sending Another Brick to the finals. Upper Cut then faced the holy wraith of Weta, God of Ugly Things, a new drum spinner out to make a name for itself. The two bots went weapon to weapon several times, sending both tumbling to opposite sides of the

arena and inverting both bots. Weta's fully invertible design allowed it to quickly rebound and attack while Upper Cut struggled to right itself. Weta won the judge's decision and went on to the finals while Upper Cut settled for third place.

The beetle weight finals featured Weta and Another Brick in a classic clash of weapon verses armor. Weta's drum ground away at Another Brick's front armor, but could gain no purchase while Another Brick's superior drive train and traction pushed Weta into one wall after another. Another Brick won the judge's decision and the first place trophy.

Meanwhile, 21 hobby weight robots clashed in a tournament dominated by big weapons and hard hits. The winner's brackets were dominated by two very different robots: Scurrie — the winner of the Franklin Cup 2009 — armed with an under cutter spinning disk and a reputation for destruction; and Zandor — armed with a compact, high speed vertical spinning disk. These two bots made their way through the brackets leaving a trail of broken bots and shattered dreams in their wakes. In the winner's bracket semi-final match, both bots came out hitting hard, but it was Scurrie that scored the first critical hit and sent Zandor down to the loser's bracket to face the big steel drum of Ntertainment.

Zandor's fight with

Ntertainment was a fast paced, hard hitting dual. Both robots scored major hits that sent sparks flying and shook the arena. As the match approached its third minute, Zandor gained the upper hand and dominated the match to the end. Zandor won the decision and the chance to face Scurrie again in the final. Ntertainment was sent home with third place. Scurrie verses Zandor part two was another hard hitting match that thrilled the crowd. In the end, Scurrie defeated Zandor again and took home the first place trophy.

Featherweight Higgins prepares for combat. Team Tech, Worcester Polytechnic Institute.



Upheaval vs Mangi. Upheaval tries to get a flip in.



Robot Conflict also features two classes of 30 lb robots: the standard feather weight combat class and NERC's own "Sportsman's class." The latter uses a special rule set designed to encourage interesting and creative designs as opposed to the all too common sight of big spinny things verses armored boxes.

This year's Sportsman's class featured 12 robots that included clamp-lifters, pneumatic flippers, overhead hammers, and other designs not normally seen in combat classes these days. Among the fan favorites was Upheaval, a big yellow pneumatic flipper that tossed its opponents (and sometimes itself) across the arena. Upheaval lost the winner's bracket semi-final to a four-wheeled bot with an articulated lifting spike named Shish-kabot.

Upheaval then faced a fast and deadly hammer bot named Mangi. Mangi's hammer was powered by an EV-Warrior Electric bike motor which is well known for its power and ruggedness. The hammer scored many hits doing visible damage to Upheaval's thin cover panel, but Mangi was unable to avoid Upheaval's pneumatic spatula and was sent head over heels across the arena countless times. Mangi lost the match and settled for third place while Upheaval moved on to face Shish-kabot again.

The Upheaval/Shish-kabot rematch was a tough fight. Shish-kabot got in a few good wall slams, but it was the high flying flipper that carried the match. Since NERC runs a true double elimination bracket, a repeat of the final match was required. For a third time, these two titans clashed in an epic battle of slams and flips. Upheaval managed to best his adversary again and take home first place.

By far, the biggest and most impressive hits of the weekend came from the feather weight combat class. Ten of these monsters tested the limits of the NERC arena. After losing a first round match, Higgins, a new drum



Spyro prepares for Sportsman rumble.



Awards ceremony (L-R): Jon Durand MC, John Pagano (Bot: Mangi; Sportsman class third place winner), and Miss Motorama 2010.

spinner built by Worcester Polytechnic Institute students, climbed all the way to the loser's bracket semi-final to face General Disarray. The General — armed with a thick vertical spinning disk — started strong, but then suffered a loss of main power about a minute into the match. Higgins then faced the spinning shell of doom known as Steel Shadow. The two bots attacked each other viciously; Higgins was almost counted out after a big hit but then sprang back to life and charged into the fray. The weapon on weapon hits sent sparks flying and created a distinctive clanging sound that filled the room. Then suddenly, Steel Shadow

stopped dead as thick blue-ish smoke poured out from under the steel shell. Steel Shadow suffered a fatal electronic failure giving the match to the underdog, Higgins. Normally, a rematch would be required to fulfill the double elimination brackets, but Steel Shadow was unable to repair the damage within the given time so the championship was given to Higgins.

Robot Conflict sponsors included FingerTech Robotics, BattlePack, Team Whyachi, Dimension Engineering, KitBots, and Industrial Plastics, Inc., in Woburn, MA. For more information about future NERC events, visit www.nerc.us. **SV**

FAIRY WEIGHT (150G [5.35OZ])

Place	Robot	Driver	Team	Home
1	Crim	Kyle Singer	Twisted Sick Robotics	Meshoppen, PA
2	Mango Farmer	Thomas Kenney	MH Robotics	Franklin, TN
3	Fright Blade	John Parsons	Slammers	Madbury, NH

ANT WEIGHT (1 LB)

Place	Robot	Driver	Team	Home
1	Maelstrom	Bryan Uber	Dark Steel Robotics	Paramus, NJ
2	Gilbert	Thomas Kenney	MH Robotics	Franklin, TN
3	Green Monster	George Hotz	Team Titanium Squirrels	Glen Rock, NJ

BEETLE WEIGHT (3 LB)

Place	Robot	Driver	Team	Home
1	Another Brick in the Wall	Simon Popecki	Slammers	Madbury, NH
2	Weta – God of Ugly Things	Andrew Smith	Team Rolling Thunder	Cary, NC
3	One Fierce Upper Cut	Gene Burbeck	Fierce Robots	Ann Arbor, MI

HOBBY WEIGHT (12 LB)

Place	Robot	Driver	Team	Home
1	Scurrie	Zachary O'Donnell	Brain Damage	State College, PA
2	Zandor	Alan Young	Mad Scientist	Freehold, NJ
3	Ntertainment	Paul Ventimiglia	Team Demolition	Wayne, NJ

SPORTSMAN (30 LB)

Place	Robot	Driver	Team	Home
1	Upheaval	Alan Young	Mad Scientist	Freehold, NJ
2	Shish-kabot	Alex Horne	Cookin' with Gas	West Chester, PA
3	Mangi	John Pagano	Half Fast Astronaut	Edison, NJ

Taking the STAIRs to Advance Robot Development

By Jeremy Kerfs

Commercial robotics centers almost entirely around the machines on assembly lines. Their precisely controlled movement repeats the same motion every time, screwing in bolts or checking for proper alignment. The vast majority of these do not have a single sensor on them to determine anything about their environment. Instead, they follow preprogrammed instructions because it's assumed they will always encounter the exact same situation to deal with.

Researchers at Stanford and Cornell are pioneering new vision algorithms and methods of teaching robots to interact more fluently with their surroundings in the hopes of creating machines that can respond to changes around them. While many robots already employ similar vision, they are usually attached to research or hobbyist work. The possibilities of mobile robots that can grasp and manipulate objects with dexterity close to that of humans would enable them to maneuver plants, inspect pipes for leaks, or clean up spills (for example) without being teleoperated.

Most of this pioneering research uses the STAIR (Stanford Artificial Intelligence Robot) as the base. This robot has a single arm that can be manipulated at several joints in different directions. At the end is a claw that can grasp even breakable objects like glassware. The robot platform includes a computer, power source, and a complete system of servos and wheels for maneuvering. The variability of the robot comes in the vision equipment that is used. Different experimentation has resulted in the addition of many cameras, lasers, and other sensors to detect motion, objects, and the environment.

The most recent achievement of this robot was opening doors and maneuvering elevators. This was not a small feat for a robot to perform, and it only came about after using a novel approach to vision. Most robots take in data from cameras that they run through programs to detect faces, walls, rough terrain, or similar objects. The robot computes this with data from the color and lighting

of various parts of the image. Other robots use a laser system in which a beam is shown over an object or area, then a camera tracks the laser beam, translating the refractions in the beam into a three-dimensional rendering.

The STAIR robot uses both of these processes in a combination that allows for extremely accurate identification of objects. The basis for this system is through the use of context. For example, if a human is presented with a traditional number pad, but the "4" is scratched off, a person will easily figure out where the "4" should be based on the position of the other digits. The robot uses a similar method except that often the numbers are intact; however, it cannot recognize them properly with its cameras.

A three-dimensional sensor can provide a basic layout of a panel of buttons and the camera will fill in the characters that it recognizes. The software on the robot must now interpolate the two data sets and use a map of what normal panels look like to create a complete view. With an understanding of where the button is that the robot wants, the arm must now push the button without hitting any object in the process.

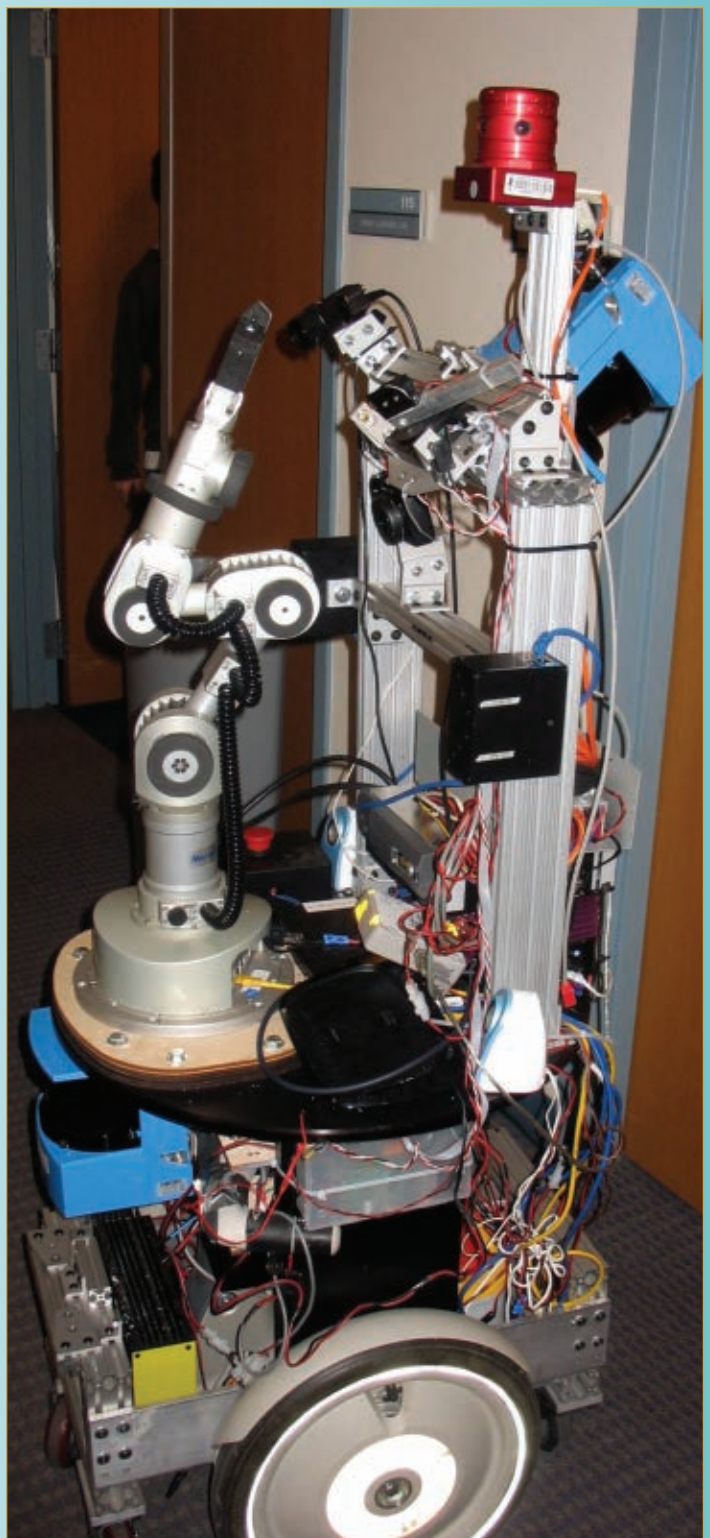
This next step is done in relation to the three-dimensional view of the environment that the robot has gathered. The arm is given precise movements to avoid objects that the scanner has identified. When the robot finally contacts the button, it has completed its objective.

How It Works

The way that the STAIR robot recognizes objects in front of it by applying the context method was researched by Morgan Quigley, Siddharth Batra, Stephen Gould, Ellen Klingbeil, Quoc Le, Ashley Wellman, and Andrew Y. Ng in their paper entitled "High-Accuracy 3D Sensing for Mobile Manipulation: Improving Object Detection and Door Opening." I had the opportunity to speak with Ellen at the Artificial Intelligence Lab in Stanford where the project is worked on. She explained a lot about the complex vision system that the robot uses. While the robot has more than five sensors related to interpreting the environment around it, the 3D scan uses the laser and single camera. Through a process called the "laser-line triangulation scheme," the camera notes the deformations in the laser as it passes over objects of varying depths.

The system creates 600 images in a single scan. The combination of these yields a three-dimensional model of points that convey the actual depth of the objects in the scene. This 3D view is then reconciled with the view from a traditional camera, adding a depth component to the pixels in the image. With a color map of the objects and their depth, the robot can implement various strategies to locate the necessary object (or button) in a cluttered environment.

If the system only used a traditional camera, the likelihood of identifying the correct object is very high. However, a number of false-positives would be encountered. The three-dimensional component eliminates



much of this because there are very few instances when an object with the coloring of a mug also happens to have 3D characteristics of a mug unless it is in fact the object itself.

The potential and efficiency of this system is demonstrated graphically in **Figure 1** where the map shown was generated completely by the robot's vision. Each red dot is where the robot stopped and scanned a desk for an object that is then highlighted with an orange circle in the yellow field of view.

The practical applications of this are immediately visible.



FIGURE 1.

For example, a robot like this could end any needless searches for a tv remote because it could systematically scan an entire house for it. It could also perform inventory searches in stores or check cars for problems.

Beyond Vision — Grasping

With an improved ability to interpret the environment around it, a robot should then be able to interact with the

world. This is another area of research on the STAIR robot. When a robot comes across a wine glass for instance, it cannot merely extend its claw and clasp it without carefully determining where it should pick the glass up from without shattering it in the process.

Most of this particular research on the STAIR robot is done at Cornell University. The researchers have found algorithms that employ the vision data to track and determine places that are crucial to holding an object. **Figure 2** shows the tag that the program gave to the mug. This area was determined to be the best place for the robot to grab — a location

remarkably similar to where most humans would hold it.

Edge detection and 3D information are both critical to determining how to clasp objects. Without prior information about an item, even the most advanced programs have difficulty determining how to pick something up because different surfaces can have varied conditions (like being slippery) or simply not a good place to hold onto.

Another seemingly simple — but surprisingly difficult —

A Glimpse of the Future

The company Willow Garage (which works closely with Stanford) makes a robot called the PR1 that has two arms and can be controlled wirelessly. The result is a teleoperated robot that is able to perform many of the tasks listed as objectives by Stanford. Humans are still in control of it, but the advances in hardware and motor control allow it to perform tasks that are incredibly complex.

Stanford has videos of the robot grabbing beers from a refrigerator, clearing a table, and tidying up a room, among other things. You can view these videos at <http://personalrobotics.stanford.edu/>. Most of the videos are sped up, but you can see that the robots have the hardware and software necessary to perform movements like humans. With the addition of vision, robots could be sweeping, vacuuming, and washing dishes.

While the PR1 is very impressive, Willow Garage is now making the PR2 as a next step. Like the PR1, it uses the ROS (Robot Operating System) to control everything from sensors to motor control. There are several advantages to having this system control the robot in that it can be treated much like a computer, and have motors and sensors work in conjunction just like sound and graphics work together on a PC.

The PR2 promises greater mobility along with easier adaptation to automated performance. Universities like Cornell and Stanford are getting PR2s to run their programs on and test the latest developments in robotic hardware. Adding the algorithms developed in the Stanford Artificial Intelligence Lab to this unique system will lead to exciting projects in the future.

Much of this new work will be completed using the unique possibilities of the ROS. Instead of loading on single programs for a robot to carry out, the robotic platform is treated as a computer. This allows for better multitasking and the ability to separate sensor input from motor control. The system is based on Linux and it can run Python and C++ programs. The distribution of special libraries based on this operating system will speed up the sharing of advances in artificial intelligence.

Combining advances in hardware and software along with the STAIR platform and its more advanced versions will be a leap forward in intelligent robots. The Stanford projects are sponsored by corporations like Google and Intel, as well as the Defense Advanced Research Projects Agency (DARPA). These groups share the same goal as the Stanford researchers to enable robots to perform more tasks without human intervention.



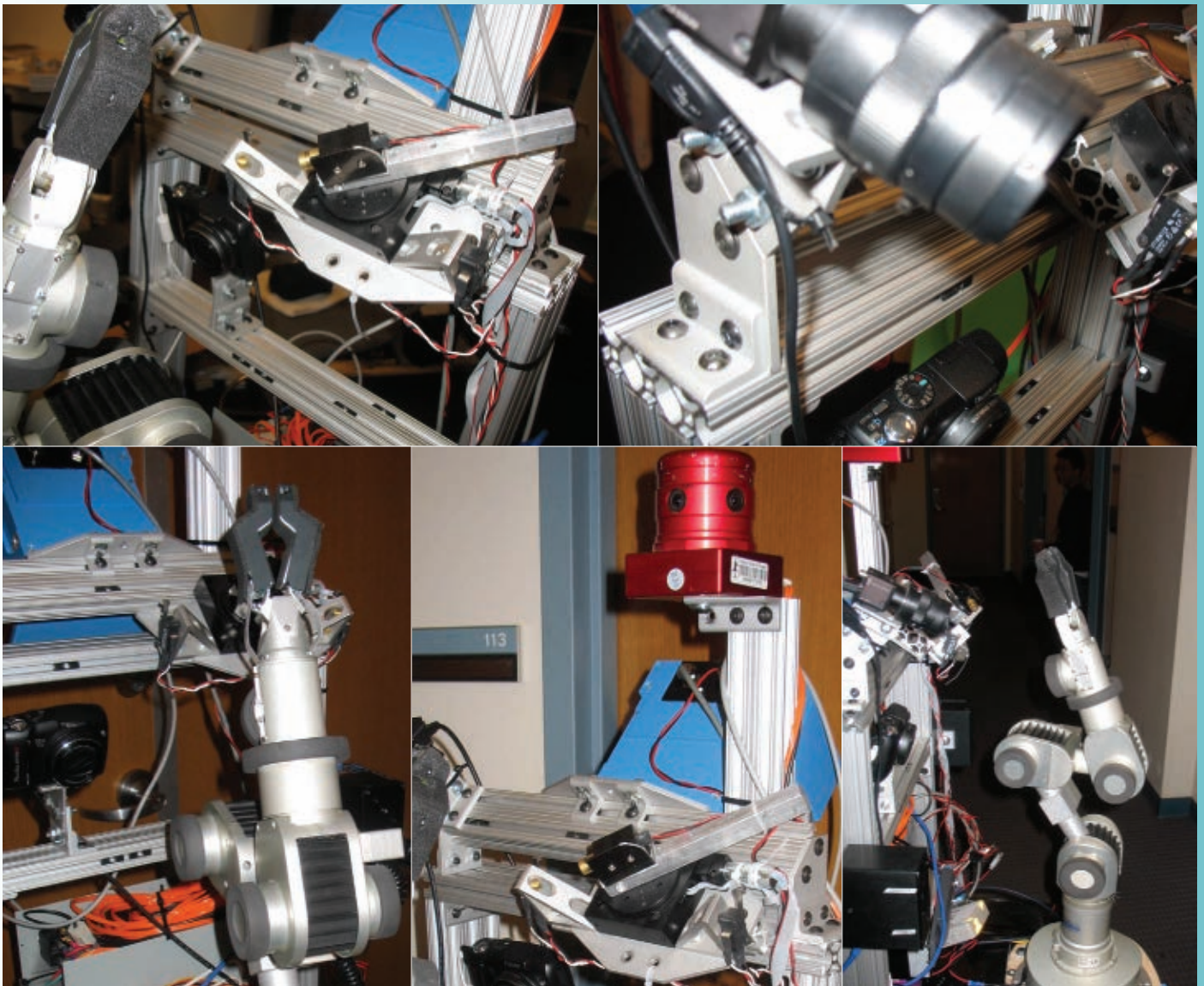
process is determining the orientation of objects. When people see a stapler, they can immediately place it in such a way that it is properly aligned for usage even if it was left on its side. For robots, this requires analysis of the edges of the object and determinations of where its center of gravity lies. Many of the same researchers at Stanford who worked on the ability of the STAIR robot to manipulate buttons also published work on recognizing objects and their orientations. Their work suggests that having a complete 3D model of an area is not sufficient enough to allow robots to interact on their own until they can understand how items are held and where the best locations are for grabbing and moving. While this work is far from complete, the potential of the research is huge. Stanford listed its own goals for a single robot that will someday be able to:

- Fetch and/or deliver items from around the house or office.
- Tidy up a room (including picking up and throwing away trash) and use the dishwasher.
- Prepare meals working in a typical kitchen.
- Use tools to assemble a bookshelf.



FIGURE 2.

With only a single arm and slow vision processing, the STAIR robot is not yet up to these tasks in either vision or object grasping. However, its rapid improvements leave open many opportunities that may make some of these tasks a reality in the very near future. **SV**



How to Prepare for – and maybe even win – a Robot Combat Event

By Pete Smith



My son, Andrew, and I make up “Team Rolling Thunder” (www.teamrollingthunder.com). We started building robots back in the UK at the height of the BBC “Robot Wars” show, and continued once we came to live in North Carolina in 2000.

We had competed and won at a local college event but found the competition much stronger in the US. Our Heavyweight *Xtreme Impax* went 0-5 in the three events we entered, and was so badly damaged at the end of the third that it was retired. We decided to move to the smaller 12 lb Hobbyweight and 30 lb Featherweight classes. We eventually did much better with our new bots and have gone on to have considerable success at several competitions. Based on our experience(s), I’d like to offer some tips on how to enter and compete at a major event, as well as give a summary of the highlights of our competition.

The Northeast Robotics Club (or NERC; www.nerc.us) hold their premier event every February as part of the larger Motorama motor sports event (www.motoramaevents.com) in Harrisburg, PA. The event is ideal for getting a broad overview of robot combat in the 30 lb and underweight classes.

The lighter classes are an ideal place to start in the sport and the Beetles or Hobbyweights in particular offer a nice blend of size and cost that make it within many people’s budgets to build an effective robot that has a chance of doing well. For your first event, it’s probably best to just convert a cheap RC car or truck and compete in the Ants or Beetles. You are not likely to win many fights, but you’ll have fun and you will learn a lot about what it takes to build (and operate) a competitive design.

Lesson 1. Get your bot finished well in advance of the event.

To be honest, we still have a problem with this. The bots always take about twice as long to repair or upgrade as you think they will. I finished getting our three entrants for Motorama ready on the very day we had to travel up to Pennsylvania. I should have gotten started a lot earlier. Completing it well before the event gives you time to make sure you are under the weight limit; thoroughly test out all the bot’s functions; and finally – and probably most importantly – it gives you time to learn to drive properly! A mediocre bot driven well will usually beat a great bot driven badly.

We set out on Friday, just before 3:00 pm. It’s about a 7-1/2 hour trip, so we arrived at after 10:00 in a very snowy (but surprisingly mild) Harrisburg and went straight to the venue at the PA State Farm Show Complex. The main Motorama events don’t start until Saturday, so you can get your car right up to a door near the pits. (This saves a long haul with all your bots, tools, and spares the next morning when this whole area is closed off.)

The Fairy and Ant weight competition is actually held on Friday, but as you can see in **Figures 1 and 2** the pits and the arena were completely deserted by the time we got there. We chose an empty pit table and stacked all our stuff on and under it. We have never had any problems with theft at these kinds of events but we do keep the obviously valuable things like our transmitter and battery chargers out of sight. We tested our 12 lb bar spinner and our two Beetles in the Ant arena which doubles as a test arena for the main event. Everything worked, so we headed to the hotel.

Event Organizers (EO) usually negotiate discounted rates with several local hotels, so always check before booking since this can save you money.

Lesson 2. Get to the venue in plenty of time.

We returned on Saturday morning around 8.00. The first advantage in doing this is that you can park reasonably close to the entrance doors. (The weather can be pretty hard this far north and the walk back to the car can seem very long!) The other advantage to getting in early is having plenty of time to get your bot(s) through Safety. You have to pass a safety check or you don’t get to compete.

The checks vary by event so it’s best to read the rules



Figure 1.

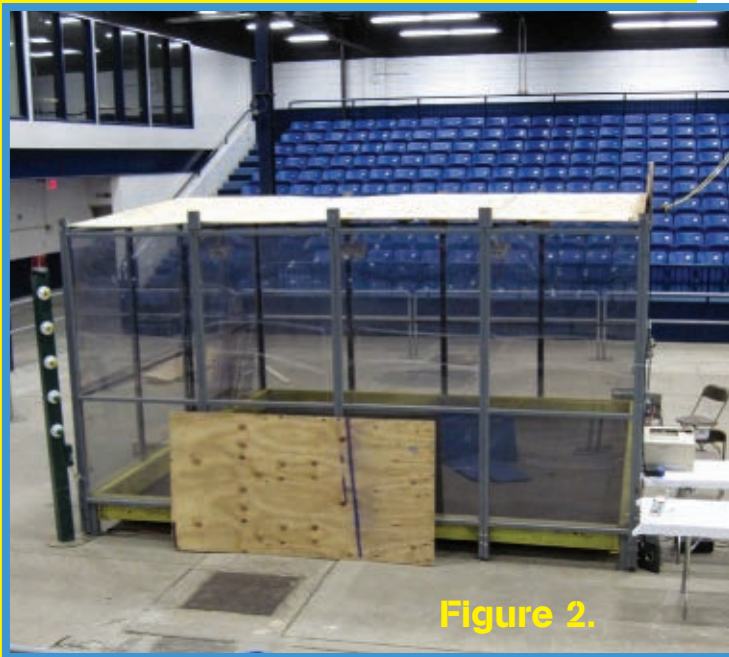


Figure 2.

carefully in advance to make sure you meet all the requirements. If you are in doubt about anything, check with the EO. Safety requirements for Motorama consist of a weight check; a check that you have a method of safely disconnecting electrical power; that any weapon can be mechanically locked; a check to see if all sharp corners are covered; and finally, a test of the bot's functions and fail safes. That last one can be the most problematic.

You spin your bot (or power up your weapon) on the spot and then while it is spinning, you turn off the TX. The drivetrain and weapon must power-down on their own within a few seconds for you to pass. If you have the right type of TX, RX, and ESCs, then this is pretty easy to achieve. However, it's not something you want to be working on to correct as the deadline rapidly approaches. Our three passed without incident.

This is a good place to add a reminder about bot safety. All weapon testing must be carried out in the test arena and all drive train testing must be either in the test arena or with the bot up on blocks so that its wheels

cannot move the bot. If you use a radio that's doesn't bind to a specific receiver, you must be in possession of the frequency clip for the frequency you are using. A new rule for this year was that all LiPo batteries must be charged outside of the bot and inside a suitable fire resistant container like a LipoSack. There is always a driver's meeting just before the fights start; make sure you attend this as there will be last minute rule and procedure changes, and there is usually a last check to make sure everybody who thinks they are competing are correctly entered into the brackets.

The actual fights didn't start until well after 11:00 am, so we had a couple of hours to get our pit table fully set up and the batteries on charge.

Lesson 3. It's best to have at least a couple sets of batteries ready at any one time.

You are guaranteed at least 20 minutes between fights, but a LiPo pack typically takes at least an hour to charge. NiCd, NiMH, and LiFe (A123) packs can be charged quicker, but if a pack is still hot from a previous fight it can easily be damaged if you try to charge it too fast. Our Beetles both use the same size of LiPo, so we had four of those; the 12 lber uses an A123 pack that we also use in our Bot Hockey bots so we had six of those.

You will probably have plenty of time to recharge at minimum a single battery early on in the competition, but if your bot is even moderately successful, the fights come hard and fast on Sunday.

Once we had everything ready, we took time to meet up with some old acquaintances and check out some of the new bots. Most builders will be happy to show you their creations and answer any questions you have. One exception to this is if they are busy trying to finish or repair their bot. In this case, leave them alone unless you are offering to help.

Four classes of bots were competing on Saturday and Sunday. There were large fields in the 3 lb Beetleweights with 19; the Hobbyweights had 21. There were 10 30 lb

Featherweights and 12 30 lb Sportsmans. This last class is designed to allow more innovative designs to compete and survive by ruling out both wedges and kinetic energy weapons.

We had entered one 12 lb Hobbyweight (*Surgical Strike*, seen on the right in **Figure 3**) which had taken first place at both the 2008 and 2009 events. I had improved the electrical wiring and made a new hardened S7 blade for it. Our second robot was my son's Beetleweight spinner *Pure Dead Brilliant* (**Figure 3**, left) which had taken a first place one year at Franklin, but had yet to do well at Motorama. Our final entry was another Beetleweight — *Weta, God of Ugly Things* (**Figure 3**, center). This was its first



Figure 3.

competition and our first drum bot. The plus side of entering so many bots is that you are guaranteed more fights and it reduces the chance that you get knocked out early on. The down side is that you are kept *really* busy both fighting and maintaining your bots.

All weight classes were very closely fought. In our first fight, *Surgical Strike* fought *It Stings* – a new drum bot from Florida. We were pretty confident going into the fight since our large horizontal blade has usually had little problems with drums in the past. We got a couple of good hits in when our bot just suddenly stopped working and was counted out. The problem turned out to be nothing more than a broken connector between the battery and the bot's wiring loom.

Lesson 4. If something doesn't seem quite right, take the time to check it out and fix it.

I recalled that the connector had not gone smoothly together when I was completing and testing the bot the day before. It had worked okay though and I had thought nothing more of it. It turns out the spring section of the Deans type connector had broken and fallen out so that the contact didn't make a reliable connection. It had worked fine in the Safety test and in the test box just before the match, but the hits had broken the connection and killed all power. A new connector quickly fixed this but it was frustrating to have lost for such a simple reason – doubly so since it turned out we had already damaged *It Stings* enough that it had to retire from the competition. The lesson of fixing things when you know something is not quite right would come back to bite us again later.

Pure Dead Brilliant's first fight was against *Misdirected Aggression*. The latter has what is probably the longest Beetleweight blade in the world and far outreached the blade on our bot. The fight started really well with *PDB* removing one of his opponent's wheels in the first hit. That did not stop *MA* from shuffling around on one wheel and with the blade still spinning dangerously. Andrew went in for a couple more hits, then kept out of range until it was apparent that the big blade on *MA* was slowing down. Andrew attacked again and after a couple more good hits, *MA* tapped out.

The effectiveness of that long blade can be seen in the damage done to *PBD* (**Figure 4**) where the cut goes all the way to the axle bearing. Nasty, but not fatal. The weapon drive system on *MA* had failed and was not repairable, so another bot was out of the competition.

Our next fight was with our new drum bot, *Weta*, *God of Ugly Things*, against horizontal blade spinner *Traumatizer*. *Weta* is actually a test bed for a new Beetleweight kit I plan to have available in the summer. It's a fairly conventional design but with one new feature. Drum bots often end up inverted – either through a hit by

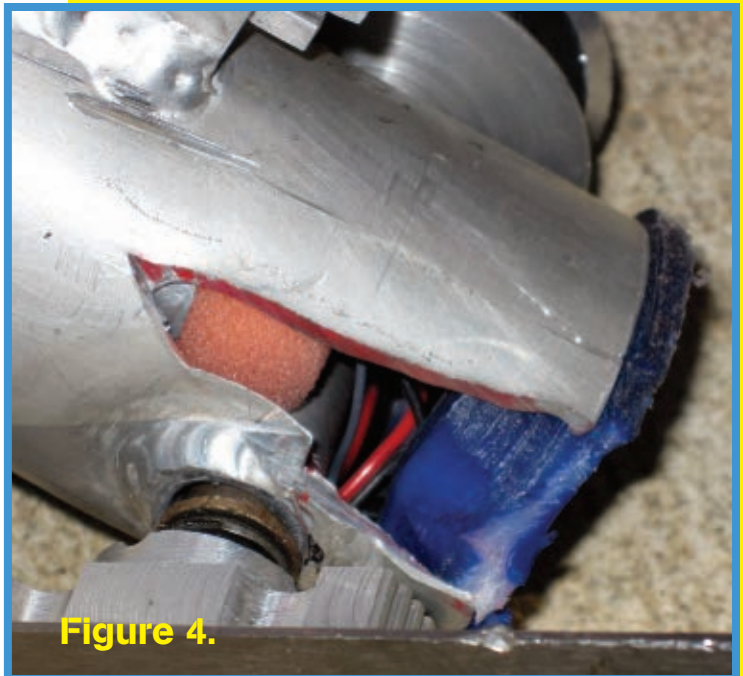


Figure 4.

an opponent or sometimes even through gyroscopic effects – while trying to turn fast (this is called “gyro dancing”). Once the bot is inverted, the drum is much less effective as it can no longer throw the opponents up into the air. *Weta* gets around this problem by using a reversible brushless speed controller that can stop and reverse the rotation of the drum in seconds. These reversible controllers are designed for RC cars and once were very expensive, but they're now available from companies like Hobby City (www.hobbycity.com) for less than \$35.

The wheels also go in the wrong direction when your bot is upside down, but that can be fixed by using a drive ESC that has an “invert” or “flip” function; just add a custom mix in a programmable radio TX or simply doing the changes in your head. The last is the way Andrew (he usually drives in the competitions) prefers to do it.

Weta's UHMW armor was unperturbed by *Traumatizer's* blade and Andrew dominated the fight to get a quick judge's decision. The bot was flipped a couple of times but he had simply reversed the drum and kept fighting.

We were up again the very next fight with *Pure Dead Brilliant* fighting *Sting*. This bot was a miniature version of the 12 lber “*It Stings*” that we had met earlier. *PDB* lost a tooth (the 10-24 mounting screws had sheared off) which unbalances the blade to the point that it is useless, and then we lost one wheel so we tapped out and *PDB* was down into the loser's brackets.

We had a break of a couple of hours which gave us time to prepare the bots for the next fights. We fitted our new S7 hardened steel blade to *Surgical Strike* since we knew we had a tough fight next against *Deranged*. *Weta* just needed a freshly charged battery, while *Pure Dead Brilliant* required more work needing a new tooth and replacing the drive motor on one side.

Figure 5.

Lesson 5. Maintenance should be a serious consideration when designing your bot.

It's no good having plenty of spares if it takes too long to use them. You need to be able to virtually rebuild your bot in about 20 minutes. If a part of the bot would take longer than that to repair, then have a spare part of the sub-assembly ready to swap out. Try to use common sized fasteners so you aren't scrambling to find the right tool or that uniquely sized screw that just fell on the floor! It's tempting when building a bot to hard-wire all the parts together as this saves both weight and space. However, this can quickly become a nightmare when you need to replace a component. Use quality connectors between each part. You can add a little tape to each connection to ensure they stay together. It's important too, that you are methodical about tracking down exactly what has failed before taking your whole bot apart. I once assumed it was an ESC that had failed when it actually had just been a loose RX signal cable. That mistake cost us placing at Motorama 2006

when we couldn't get the bot back together in time. You can use a spare motor to quickly test if the electronics are all right; an RC servo provides a good way of testing the RX side.

Our fights resumed with *Pure Dead Brilliant* taking on *Traumatizer* — the bot *Weta* had defeated earlier. This fight was much closer. *Traumatizer's* blade failed early on, then a little later the pulley on the weapon motor on *PDB* came apart, so our blade stopped as well. The fight descended into a pushing fight and we got what must have been a close judge's decision.

Next up was *Weta* against *Sting*. The two drum bots went at it head to head with *Sting* probably getting the better of the fight, but the judges gave the fight to *Weta*. This was a nice surprise for us.

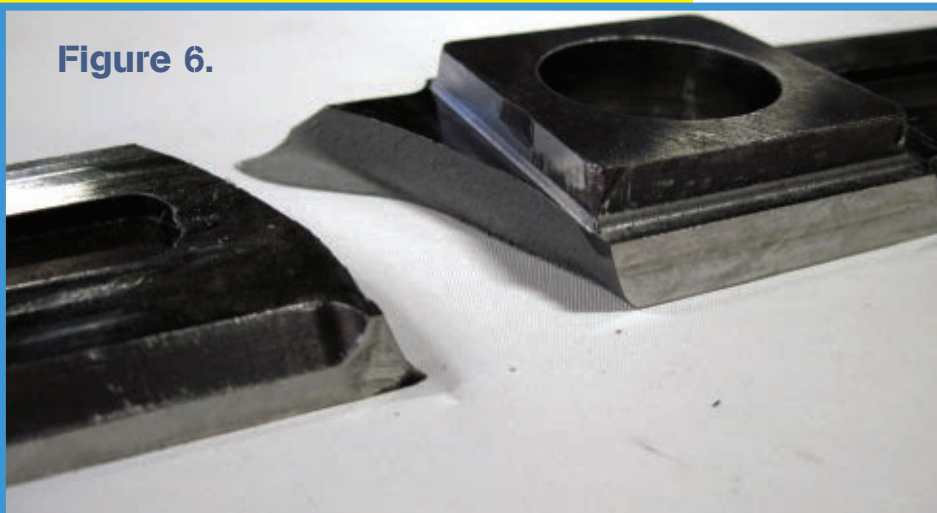
Lesson 6. The Judge's decision is final, so it's pointless to complain.

It does reinforce an important point, however. Defeat the opponent within the three minutes and you'll never have to worry about the judges! We have lost fights we were sure we had won, and now had won when we thought we had lost. Judging is difficult and how you see a fight may not be the way your opponent or the judges see it. If a decision goes against you, you be gracious about it. Sooner or later it all equalizes out.

Our last fight of the day was my favorite of the whole competition. *Surgical Strike* versus *Deranged*. We had a lucky victory over *Deranged* at Franklin in '09 but I was under no illusions that it would be that easy again. We had fitted our new hardened S7 blade and decided to go all out — no holds barred — right from the start. Bright titanium sparks flew as our blade hit our opponent's armor and we quickly cut into his underside and hit his LiPo battery. Large

quantities of smoke and puffs of flames started billowing from *Deranged* but he kept on fighting (**Figure 5**). In his last hit, he snapped our blade in two! The fight was then stopped by the judges under the new "if your bot catches fire you lose" rule, so *SS* had won by a technical knockout. The blade had been over-hardened to RC55 when RC40–45 would have been better. The clean snap from a stress concentrator can be seen in **Figure 6**.

That was our last fight of the day. We saw a few of the other fights, but *Sloth* versus *Agent 3.5* in

Figure 6.

the 30 lb Featherweights and *Upheaval's* fights in the Sportsman class made an impression on us and the rest of the large audience. We repaired the damage to Surgical Strike (refitted the original titanium blade and replaced a rough sounding gearbox), tidied up our pits area, then headed out.

We arrived bright and early again on Sunday. The fights were scheduled to start at 10:00 am. All of our bots were still in and the fights would be coming fast and furious.

Several teams didn't get back in time and had forced postponements of their fights. You only get one postponement per bot per day, and losing that one chance of an extra 20 minutes for repair time can easily cost you a win later.

Lesson 7. Know when you are scheduled to fight and be ready and on deck in plenty of time.

The event organizers have only a finite time to get all the fights in, so they should not have to go hunting for you. Remember, the event is entertainment for the paying public and an empty arena is not very exciting! The brackets are usually available on a couple of screens beside the judges and are posted on WiFi throughout the event. It is your responsibility to know when you are fighting and to be ready. The organizers do not want to have to make people forfeit matches for this reason, but don't be surprised if you find you "lose" a match because you weren't where you were supposed to be.

Our first fight on Sunday was *Pure Dead Brilliant* versus *Revenge of Dr. Super Brain*. Our opponent is a lifter/flipper but it proved no match for *PDB* and after being knocked around for a while, he tapped out. *Surgical Strike* also had a relatively easy fight against the wedge *Acute Pain*. We ran over the wedge a few times until *SS* ripped off the front edge of it. *Acute Pain* proved tougher than expected, we did enough to get the win by a judge's decision.

I thought the next fight would be easy, too. *Weta* was to fight a brick bot appropriately called *Another Brick in the Wall*. The problem was that *Weta* could not get a bite on *ABITW's* front armor and that (combined with his excellent driving and powerful drive train) meant a judge's win for our opponent. *Weta* was now down into the loser's brackets.

Surgical Strike was up again next against the full body spinner *Sonic*. This was possibly the shortest fight of the weekend, with only about five seconds of action. Both bots spun up and *SS* drove over and delivered one big hit. The bots flew in opposite directions. *Sonic* was upside down with its shell badly dented, while *SS* was upright but the weapon was disabled and drive was only available on one wheel. *Sonic* was counted out and we had won, but at great cost. The G forces must have been

tremendous when the teeth on the bots met. The big brushless motor that powered the weapon was bent (**Figure 7**), and both drive motors were damaged and needed replacement. We had a second brushless so that was not a problem, but we had already used up one of our two spare drive motors so we were one short. Luckily, another competitor had a spare we could buy and I managed to assemble one good one out of the parts of the three damaged ones. The blade had been driven down the shaft a bit and we had to hammer it back up the shaft to allow it to turn freely. This has been a recurring problem with our titanium blade. We have been using it since 2006, so it's a little bent and the hole used for mounting it is just a tiny bit too big for the keyless bushing we use to secure it to the shaft. The new *S7* blade was the solution to this problem but it had been over-hardened and had snapped fighting *Deranged*, so we had to keep using the old blade.

Pure Dead Brilliant then fought *Son of Thump*. This bot had interchangeable weapons and had swapped its wedge/lifter for the drum from *Sting* and the fight went the same way as the first. *PDB* lost a tooth and then a wheel, so we tapped out. *PDB* was out of the competition.

We quickly got our revenge, however. Next up was *Weta* against *Sting*. The two drum bots went at it head to head with *Sting* getting the better of the fight initially until *Weta* dislodged and cut *Sting's* power up-link to get the knock out.

Surgical Strike's last fight was up against the excellent drum bot *Ntertainment*. I had pretty high hopes of beating him this time as the bot seemed to be having some problems in previous fights, but it was not to be. An early hit knocked our blade down the shaft again jamming it, and a second hit wrecked the weapon's drive motor (again!), so we tapped out.

Weta was our last bot still in the competition and our next two fights were to be against Gene Burbeck's *One*

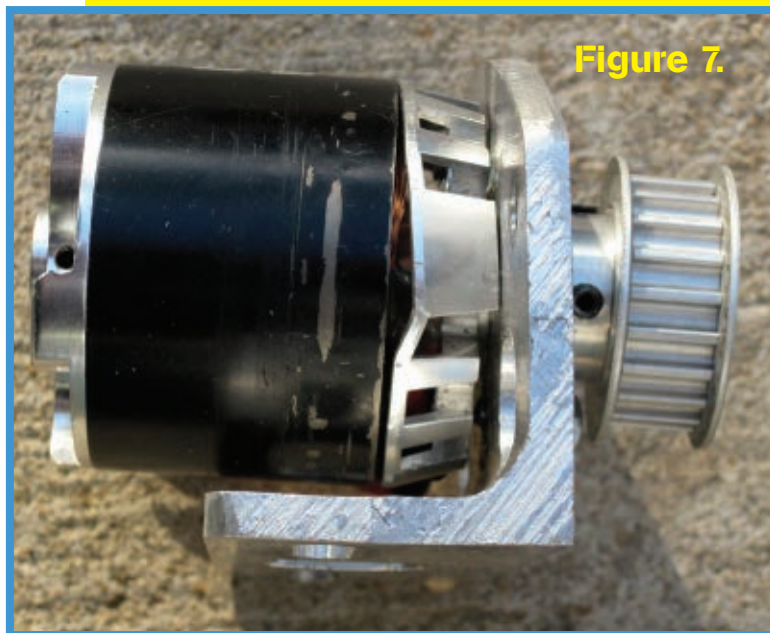


Figure 7.

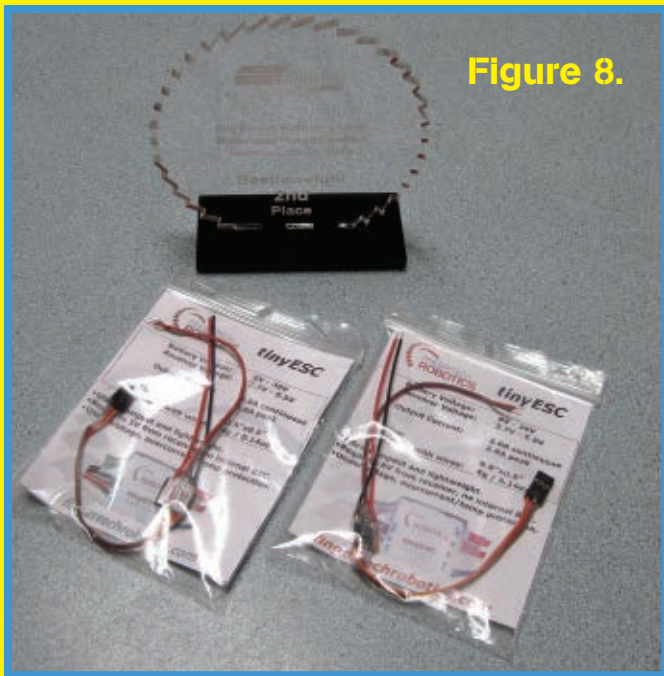


Figure 8.

Fierce Round House and *One Fierce Upper Cut*. The first bot is machined from a solid block of UHMW and has a tough undercutting blade. We definitely got the worst of the damage, losing two teeth and having the drum's axle start to come out, but *Weta* had been more aggressive and that seemed to have swung the judge's decision in our favor. *Weta* was starting to look a little battered and since we could only fit two teeth to the drum now, it was a little out of balance. This didn't matter though against *One Fierce Upper Cut* where we switched off flipping each other over. This was no problem for *Weta* with its full invertability and reversible drum but *OFUC* was virtually helpless when

Action Photos by Brian Benson; www.bensonpv.com

If your bot has been knocked out early in the competition, there are still chances to fight again. There were only a few semi and final matches still to be decided in the various weight classes and since each bot is allowed at least 20 minutes between fights (and allowed to postpone for an additional 20 if needed), there are often gaps which need to be filled in order to keep the crowds entertained. This is where grudge matches and rumbles come into play. Grudge matches are where two teams can re-fight previous matches or two bots that have never met before can prove who really is the best. A rumble is where all the bots that were knocked out earlier get into the arena together for one giant, five minute melee. There are no prizes for grudges or rumbles, but they are often some of the best parts of an event. The pressure is off, so you can just have fun. Unfortunately, *Surgical Strike* and *Pure Dead Brilliant* were already out and not readily repairable so we missed out on the fun this year.

overturned. Andrew flipped him back once but the second time he got stuck in a corner and was counted out. *Weta* was in the finals in its very first event!

Weta had to fight *Another Brick in the Wall* again for first place but the fight was a repeat of the previous battle, so *ABITW* won easily. We were still pleased with a second place finish with our first drum's first competition.

And the Winners Are ...

The fights were completed by early Sunday evening and we had won a very smart trophy and a couple of Ant sized ESCs (Figure 8) which had been donated by Fingertech Robotics (www.fingertechrobotics.com). Prizes are always cool, but keep in mind they'll have a value far less than the cost of competing, so don't enter with any expectations of ending up ahead financially. You will, however, have an experience to remember which is well worth the cost.

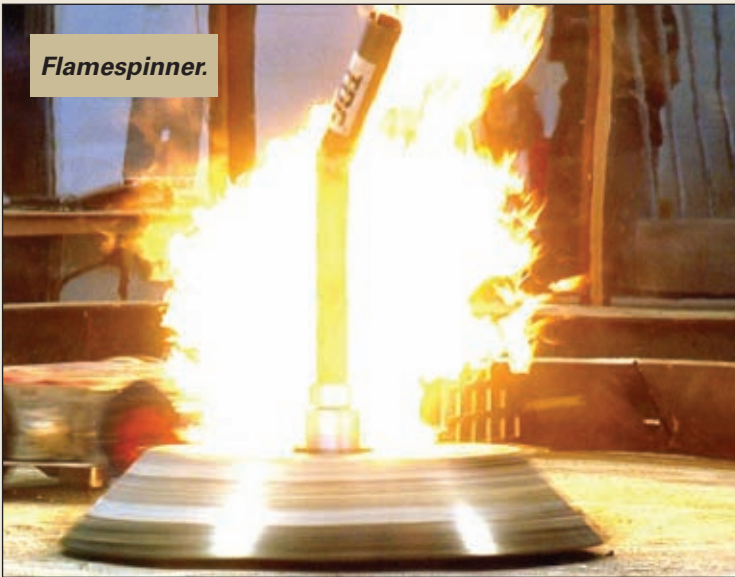
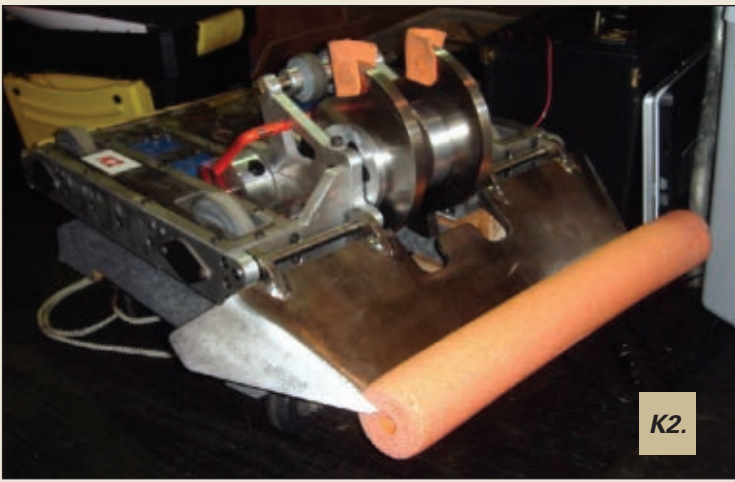
Motorama has become our favorite event and the one we make sure we don't miss each year. It also makes an ideal first event as the friendly competition and organizers will make any newcomer feel right at home. You might not win any fights the first year, but if you watch and learn what works and what doesn't you'll soon be building bots that are competitive. With a little luck and a lot of hard work, you might even take first place. **SV**



So You Want To Build A ComBot

Part 2

By Greg Intermaggio



Last time, we talked about the individual components that go into a ComBot and how they work. We then designed a chassis for our featherweight ComBot. In this second installment, we'll be putting it all together, and turning that bucket of parts into a working ComBot!

GETTING READY TO GET BUILDING

As discussed in Part One in the April issue, you should have a solid design in mind for your 30-pound ComBot. Here's a few more design tips and a quick review before we start building:

- Avoid overly complex chassis designs so when the bot gets beat up, it doesn't take long to fix. (This also means avoid welding — use bolts. Lots of bolts.)
- Keep in mind that you'll need easy access to the inside of the bot. Try to minimize the number of connections on your main plate, without sacrificing too much integrity.
- Start by designing an armored chassis. Once you have a working wedge-bot, *then* add an active weapon.
- Remember that the wheels you buy won't attach naturally to your motor shaft. Double-check that you're buying the appropriate accessories to make a strong connection.
- Make sure EVERYTHING fits inside your chassis (don't forget the batteries!).
- Double and triple check that you understand how all the components work together, and that your design incorporates all of them prior to ordering parts.
- Remember to pair your LED indicator light with a compatible resistor, and put the LED somewhere visible and protected.



Touro.

When you're ready to get building, you can find a complete list of parts and where to order them in Appendix A, which is available online at www.servomagazine.com.

PARTS CHECK & WEIGH-IN

Once all the parts for your ComBot have arrived, the first thing to do is make sure nothing is missing. Go through your laundry list and be sure to contact the appropriate supplier if anything isn't right. After this check, it's time to weigh everything in and make sure the parts are under the 30 pound weight limit. Plop everything that will go into the robot on a scale.

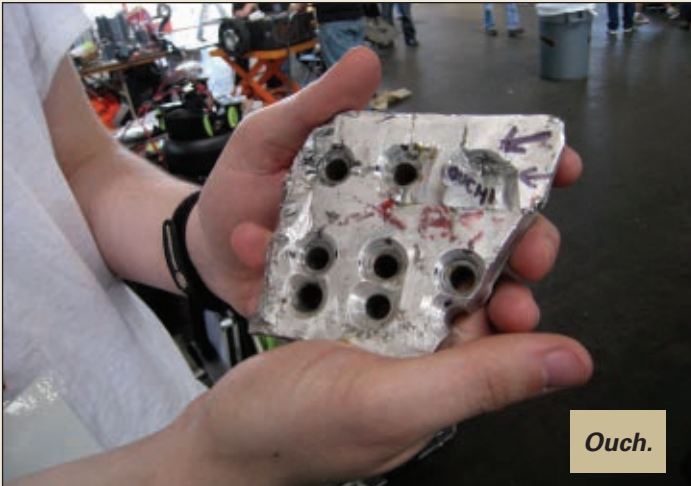
BotTip: Losing weight on a ComBot can be easy! Start by looking at the structural components. Is a solid 1/4" steel plate really necessary? You can try using a mill or a drill to thin out lower impact areas; use hollow pipes instead of solid; or any number of other possibilities along these lines to lose the fat.

TRYING THE BOT ON FOR SIZE

The next thing to do is put your ComBot together to make sure everything fits together. Lay the base plate out and put the components on top of it in the configuration of your design. Now, eyeball the other structural components and determine if your measurements were correct. If there are any issues with components not fitting or if the armor doesn't look like it'll squeeze onto the frame the way it's supposed to, find an alternative now — before you go through the trouble of trying to put everything together for real.



BR6000.



Ouch.

PARTS LIST

- Armor and Chassis
- Speed Controllers
- Motors
- Wheels and Hubs
- Batteries
- Radio Control Transmitter
- Radio Control Receiver
- Fuse
- LED Indicator Light and Resistor
- Master Power Switch
- Wires

TOOLS REQUIRED

- Eye Protection
- Measuring and Marking Tools
- Saw (Most saws capable of cutting metal will do — use what you're comfortable with!)
- Screwdriver
- Hammer
- Wrenches or Vice Grips



ComBot repairs.

MARKING, DRILLING, AND PUTTING IT ALL TOGETHER

After confirming that everything is going to fit, it's time to mark out the chassis. It's easiest to mark out component locations on a square base plate, so if you're intending to cut the base plate into a different shape, you should map it out beforehand. Start by marking where the moving parts like motors and wheels will go, since they'll require the most precision. Once that's finished, mark out the locations for the other components. Take care and be as accurate as possible when doing this. Remember, "measure twice, cut once."

When you're finished marking out your bot, it's time to cut and drill. Again, take care in your machining to be as accurate as possible. Any mistakes could mean buying a new metal plate if you can't make it all fit. After getting your armor and chassis in shape, take a deep breath and then screw/bolt all the parts into place (leave the wiring disconnected).

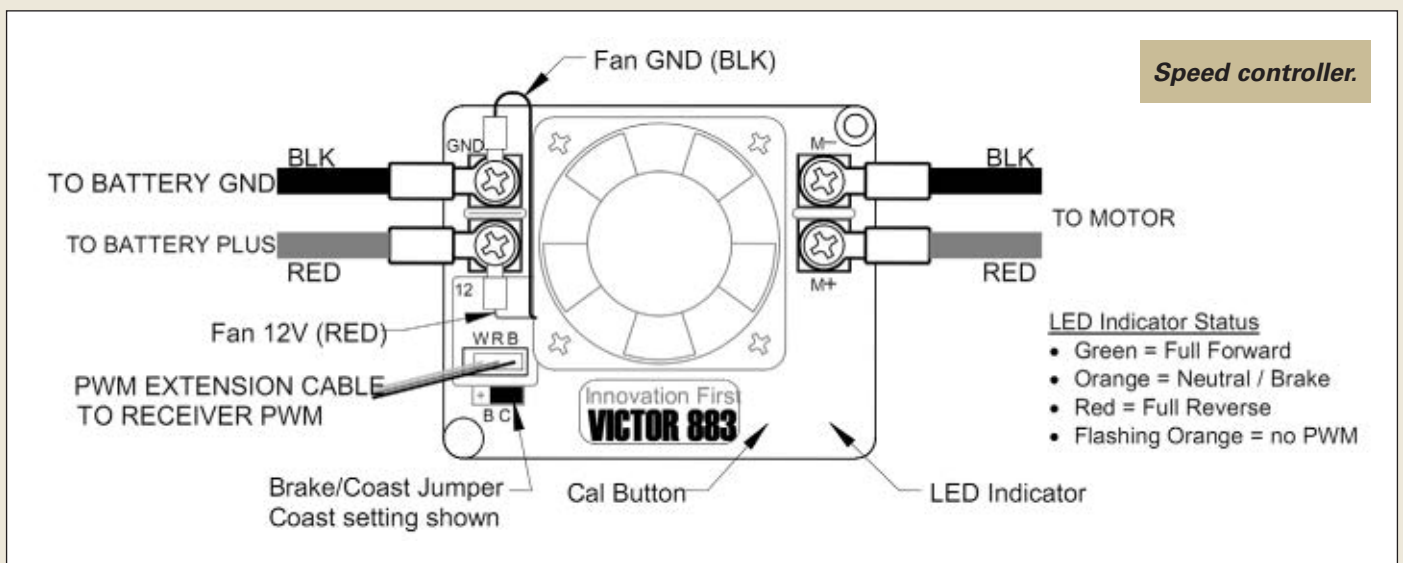
PLUG IT IN, PLUG IT IN

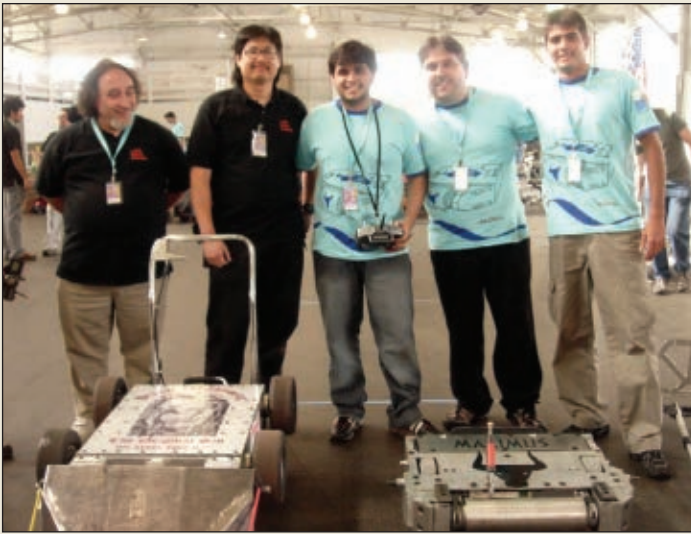
It's time to test your design. Start by soldering wires to the negative and positive terminals on one of your motors, and attaching them to the M- and M+ terminals on one of your speed controllers. Repeat the same process with your other speed controller and motor pair. Now, connect the signal boosting cables from the speed controllers to the BR6000 receiver, taking care to plug the black prong into the "-" terminal and the red prong into the "+" terminal. The white prong then goes into the final terminal. You'll want to put your left drive motor in the THR (Throttle) slot and your right drive motor in the ELE (Elevator) slot. Now, put your LED in place if you haven't done so already, and solder the resistor to it. Solder a medium length black wire to the negative side of the LED/resistor assembly and a red wire to the positive side. Finally, attach a negative wire to the GND- terminal of each speed controller, and a positive wire to the GND+ terminals.



Next, unravel the wires for the 24V fans on each speed controller and solder them to the ends of the wires connected to that speed controller's GND ports (negative to negative, positive to positive). Put your fuse in the fuse holder and wire it to the negative terminal of your battery

BotTip: When soldering, make sure you have a mechanical connection between the parts you're trying to solder. (That means a connection that would hold together without the solder.) Use solder to reinforce the connection, not to make it.





Late Night Racing and RioBotz.

connector. Connect your master power switch to the other end of the fuse. The (currently) free wire coming out of the master power switch will be the main connection point for your negative wires. The positive cable coming from your battery connector will be the main connection point for the positive wires.

Here's what should plug where:

Negative Side:

GND- cables from speed controllers
Negative 24V fan cables from speed controllers
Negative side of the LED circuit

Positive Side:

GND+ cables from speed controllers
Positive 24V fan cables from speed controllers
Positive side of the LED circuit

RECAP

(DO NOT CONNECT YOUR BATTERY YET!)

The battery plugs into the battery connectors. Power flows from the negative terminal of the battery through the fuse (which will break if you have a short circuit), and then through the master power switch. If the master power switch is on, electricity will flow through the switch into the speed controllers (which, in turn, power the motors), then into the fans (which cool the speed controllers), and finally into the LED circuit, indicating that the robot is powered on. This is the main electrical system of the bot, but there's still a bit more to do.

Next up, insulate the hell out of everything. Look at EVERY connection and make sure that it has the correct polarity, the wire is going to the right place, the connection is VERY secure, and that the connection is nowhere near anything that might short it out. (Remember: This robot could potentially be thrown in the air and come crashing down onto a hardened steel floor. Make your connections secure the first time.)

BINDING THE TRANSMITTER AND RECEIVER

Binding is the process of linking the transmitter and receiver so they talk to each other and are able to filter out interference. To bind your transmitter to your receiver, follow these steps:

- 1) Disconnect the receiver cables from the speed controllers.
- 2) Insert the bind plug into the "BAT" terminal.
- 3) Insert the receiver battery into any free port on the receiver.
- 4) At this point, the LED on the receiver should be flashing which means it's ready to bind. Pull and hold the trainer switch in the upper left-hand corner of the transmitter as you switch on the transmitter.
- 5) The flashing LED on the receiver should go solid after a few seconds, meaning the binding process is complete. Now, unplug the bind plug and battery, and turn off the transmitter.

FINAL SAFETY CHECKS

It's almost time to power the bot on for the first time. Go through every electrical connection at least twice more, making sure that every wire is where it should be, everything is secure, and no wires are mis-colored. Ideally, you should perform your first test outside, since there is always a possibility of a battery exploding (and you want it to be as far away from anything flammable as possible). When you're ready, here's how to power up your robot:

- 1) Raise the bot off the floor so no moving parts can touch anything. Make sure it's secure — your ComBot will have some torque and may jerk around despite the wheels not touching the ground.
- 2) Turn on the transmitter first.
- 3) Plug in the receiver battery, and wait for the solid light indicating communication.
- 4) With the master power switch OFF, plug your battery into your battery connectors.
- 5) Make sure your transmitter joysticks are in a neutral position.

DOCUMENTATION

IFI VICTOR 883 DOCUMENTATION:
www.ifirobotics.com/docs/ifi-v883-v885-users-manual-9-25-06.pdf

SPEKTRUM DX6I DOCUMENTATION:
www.spektrumrc.com/ProdInfo/Files/SPM6600_DX6i_Manual-LoRes.pdf

SPEKTRUM BR6000 DOCUMENTATION:
www.spektrumrc.com/ProdInfo/Files/Instructions_for_AR6000R_Robot_Receiver.pdf

- 6) The moment of truth: Have one person with both hands on the controller and another person to turn the switch. Turn on the master power switch — you should see power to the speed controllers and the fans almost immediately.
- 7) A solid green light on each speed controller indicates that they are communicating properly with the receiver. If you have solid green, try adjusting the joysticks, and watch your wheels spin!

IF SOMETHING GOES WRONG

Immediately turn off your master power switch and disconnect your battery completely from the circuit. Determine whether the problem was an electrical failure or a communication issue, and where the problem originated. Check all the related connections, and consult the manuals for the involved components (see the **sidebar** on Documentation). Patch it up and keep trying until you have a working ComBot!

NEXT STEPS

Now that you have a working ComBot, the next step is practice. Dave Calkins — a former BattleBots judge, prolific robot builder, and one of the founders of the ComBots competition — stresses that you should drive your robot every day. Two hours a day. Get over 100 hours of practice

before competing in your first event. He also wrote an excellent article called “Judge Dave’s Guide to Winning,” geared towards anyone in robot combat. It’s also included in the download package on the *SERVO* website.

Here’s what Calkins has to say in a nutshell:

- 1) Go to the competition with a 100% working robot.
- 2) Practice. Practice again. Practice MORE!
- 3) Have some sort of self-righting mechanism.
- 4) Test your armor by beating the hell out of it with a sledgehammer and seeing if the bot still runs.
- 5) Have an active weapon.
- 6) Test your weapon by attacking a realistic opponent (a hunk of steel, not a teddy bear).
- 7) Watch as many competitions as you can, and learn from them.
- 8) Bring good batteries that will last a full match; bring at least one full set of spares.
- 9) Don’t let the match come to a judge’s decision. Go for a knockout.
- 10) Read and know the rulebook of each competition you go to BEFOREHAND.

If you have any questions about ComBots, electronics, or anything covered in this article, you can contact me at g.intermaggio@gmail.com.

I hope you’ll be inspired to build your own comBot. See you in the pits! **SV**



What will you choose...

Good or Evil?

Pick your design at:

www.solarbotics.com/beetlebot

Either way, you’ll end up with an award-winning design that was featured on Makezine.com and Instructables.com. The speedy BeetleBot bounces smartly off obstacles. It assembles quickly with just a screwdriver, ideal for everyone!

SKU: KJB

Price: ~~\$39.95~~ **\$34.95**

Special introductory price!



SOLARBOTICS[®]

www.solarbotics.com



1-866-276-2687

PS- Cute or cursed? Nice or Nasty? Benevolent or beastly? Peachy or pernicious? Incandescent or irrigation? Antelope or absurd?

Go XBee-PRO With Your Robot Control

Many of you are monthly readers of both *SERVO* and *Nuts & Volts* magazines. For those of you that are *SERVO*-only readers, there are things going on over in *Nuts & Volts* that may be of interest to you. We've been working with a 16-bit PIC24F general-purpose design in the *NV* Design Cycle column that is perfectly suited for robotic and machine control work. This month, we're going to apply that 16-bit Design Cycle technology to *SERVO*-oriented robotic control applications.

By Fred Eady

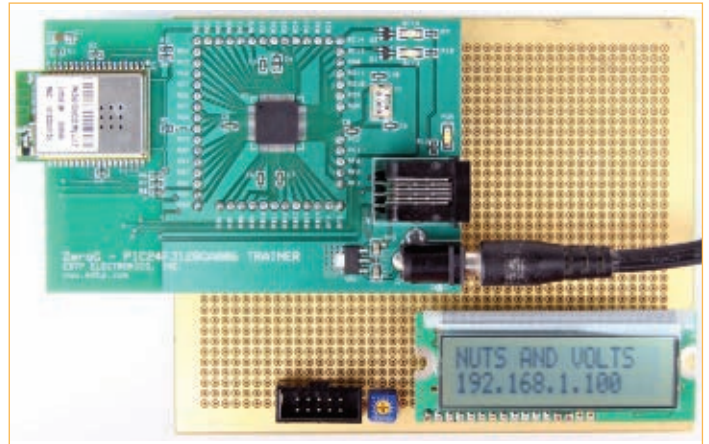


PHOTO 1. This is a shot of the ZeroG - PIC24FJ128GA006 Trainer loaded onto a Twin Industries 8100-45-LF prototyping board. A LCD and serial port were recently added to the system over in *Nuts & Volts*.

The *Nuts & Volts* Hardware

The PIC24FJ128GA006-based hardware shown in **Photo 1** speaks for itself. A dedicated ZeroG ZG2100M Wi-Fi module is hard-wired into the PIC along with a 2x16 LCD and a serial port. The 10-pin header is the RS-232 port physical attachment point while the 10K potentiometer to its right is the contrast adjustment for the LCD.

The PIC24FJ128GA006 clocks at 32 MHz using an 8 MHz crystal and the internal 4X PLL (Phase Locked Loop). A five-volt power supply feeds the Microchip TC1262-3.3 LDO (Low Drop Out) voltage regulator. I've pinned the printed circuit board's header pads with standard 0.1 inch pitch header pins. The breadboard is socketed to accept the header pins. Both five volt and 3.3 volt power are available at the breadboard level.

SERVO Modifications

The *NV* implementation works well and is based on the free Microchip TCP/IP stack which drives the ZeroG ZG2100M Wi-Fi module. In our *SERVO* implementation, we're going to replace the Wi-Fi module with an XBee-PRO 802.15.4 radio. The XBee-PRO RF modules are designed to be part of low-cost, low-power wireless sensor networks.

Drawing a peak transmit current of 295 mA, the XBee-PRO can attain a range of up to 300 feet indoors and up to one mile outdoors. To give you a perspective of the XBee-PRO's output power, remember the walkie-talkies you had as a kid? Well, they transmitted voice at 100 mW. The XBee-PRO transmits data at 50 mW. The XBee-PRO RF module requires 45 mA receiving and less than 10 μ A when its eyes are closed. We're going to use the RF module's out-of-the-box peer-to-peer communications method. However, these modules can be configured for point-to-multipoint and point-to-point operation. A mesh network of XBee-PRO RF modules is self-routing, self-healing, and fault tolerant. These tiny networks are capable of delivering RF data rates of 250 Kbps. We're also going to upgrade the project's microcontroller motor with the new 32-bit PIC32MX795F512H. This PIC is pin-compatible with the PIC24FJ128GA006 with the exception of the PIC32MX795F512H's USB I/O interface. If desired, we could actually run the 32-bit *SERVO* configuration with the Wi-Fi module since the Microchip TCP/IP stack supports the 32-bit PIC32MX795F512H and ZeroG Wi-Fi combination.

The XBee-PRO RF Module

The XBee-PRO communicates with the

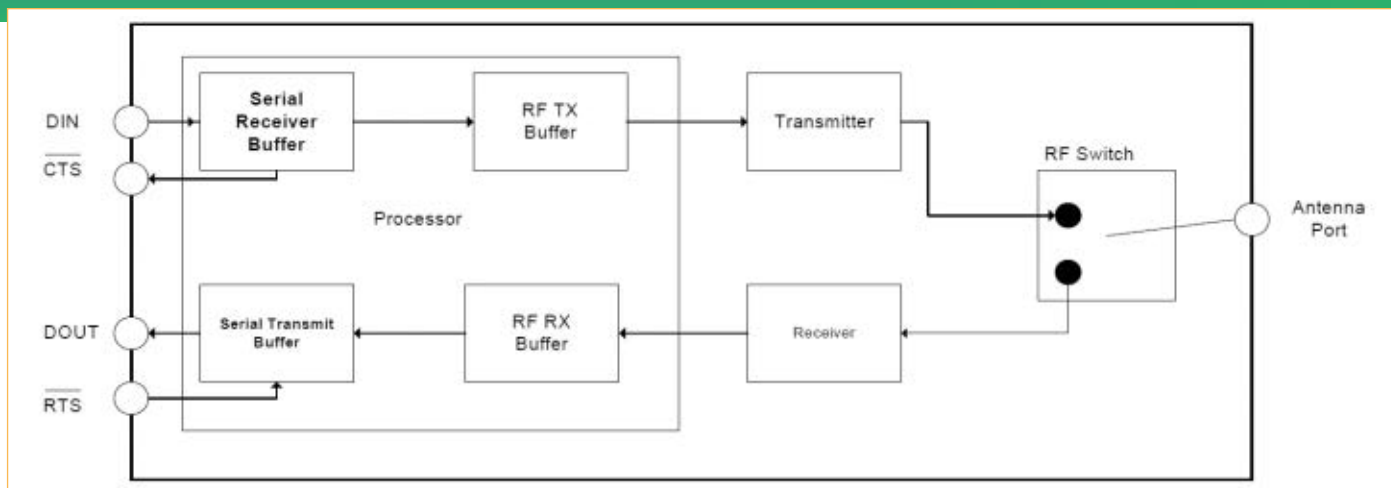


FIGURE 1. Note that the XBee-PRO uses double buffering in addition to flow control to throttle incoming and outgoing data.

PIC32MX795F512H using one of its many UART interfaces. CMOS logic levels at the XBee-PRO serial interface eliminate the need for RS-232 level translators such as the ST3232. A logical low pulse signals a START bit with the eight-bit data packet ending with a logically high STOP bit.

The XBee-PRO datasheet states that the RF data rate is 250 Kbps. However, the XBee-PRO serial interface data rate ranges between 1,200 bps and 1 Mbps. The serial I/O to RF I/O data rate disparity is obvious to the most casual observer. If the RF data rate is capped at 250 Kbps, an uncompressed 1 Mbps serial stream cannot be transmitted or received without some trickery. It can come close to being done if you use transmit and receive buffers in combination with flow control. The XBee-PRO stores incoming serial data from the host UART in an internal serial receiver buffer until the data can be processed out through the XBee-PRO's transmitter.

If the incoming serial data looks as if it will overflow the serial receiver buffer, the active-low CTS (Clear To Send) signal is deasserted by the XBee-PRO until the receive buffer is out of danger of overflowing. The active-low CTS signal is used to signal the sending UART to cease sending data when the XBee-PRO's serial receiver buffer is 17 bytes away from overflowing. When the serial receiver buffer is drained to contain at least 34 free bytes, CTS is reasserted and serial data transfer from the host can resume.

The receive serial data flow process works in a similar fashion to the transmit process and can be controlled by the host using the active-low RTS (Request To Send) signal. No incoming RF data is allowed to be processed out of the XBee-PRO's serial transmit buffer into the host UART if RTS is deasserted.

RTS and/or CTS flow control is not mandatory. However, if you wish to invoke flow control, heed the warnings found in the XBee-PRO datasheet. For instance, don't hold RTS inactive long enough to overrun the buffer, as a buffer overrun tosses the data in the buffer into the spittoon. Fred's First Rule of Embedded Computing applies here and states that in the world of embedded computing, nothing is free. Thus, after all of the overhead is accounted for, the actual maximum data throughput for an XBee-PRO

maxes out at 35 Kbps. As far as the mechanics of the XBee-PRO's flow control and buffer scheme are concerned, **Figure 1** says it all without saying a word.

It's very easy to get started with the XBee-PRO hardware you see in **Photo 2** as the modules have the ability to operate transparently. Transparent operation does not use RTS/CTS flow control and does not operate in a mesh networking topology. In transparent operation, every UART-generated serial bit that enters the module's DIN interface is routed directly to the buffers for transmission. Conversely, all incoming RF-delivered bits are immediately buffered for expulsion as serial data from the XBee-PRO's DOUT interface. The XBee-PRO knows to move the buffered data by way of a packetization timeout period, a buffer full condition, or a command mode sequence.

XBee-PRO modules are also capable of operating under the control of an API (Application Program Interface). As the term API infers, a module operating in API mode is logically attached to the host application program. API mode is commonly used under the following circumstances:

- When RF data must be sent to multiple destinations.
- When remote configuration commands need to be sent to manage network devices.
- When I/O samples need to be received from remote network devices.

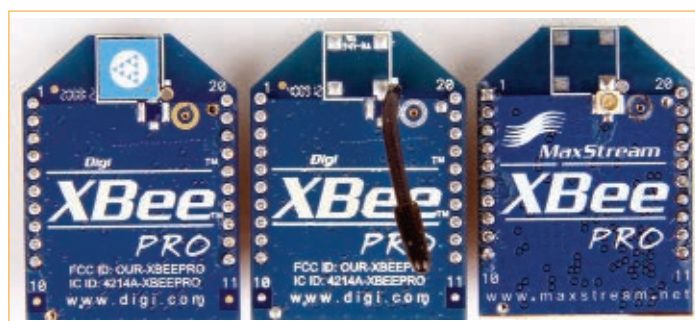


PHOTO 2. There are a bunch of pins on these puppies. However, we only need to use four of them to get on the air. I've dropped three of the XBee-PRO module variants onto the backdrop in this shot.

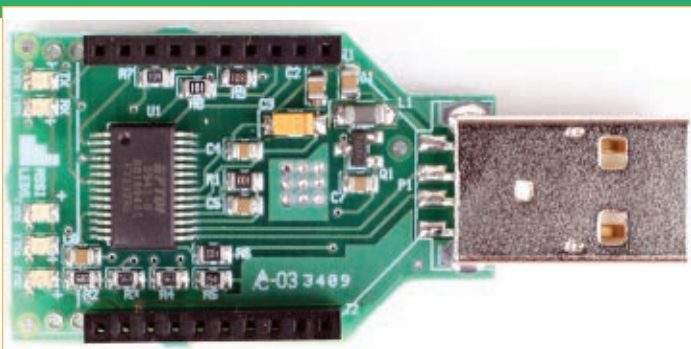


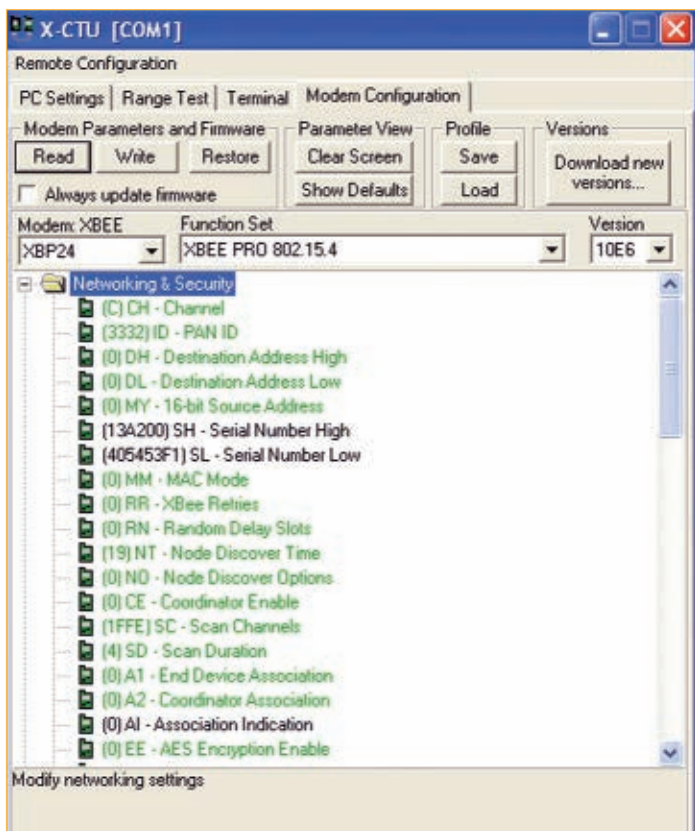
PHOTO 3. Once you understand how the FTDI parts work, you'll enjoy browsing the schematic diagram of this device. The quad comparator module design that drives the USB-XBEE-DONGLE CARRIER's signal-strength LEDs is very clever. The voltage regulator is on the hidden side of the board.

- When multiple remote devices need to be identified by their unique address.
- When mesh networking is employed (ZigBee).

As our little peer-to-peer "network" will consist of only two nodes, we'll keep it simple and operate transparently.

The Personal Computer Node

The PC XBee-PRO transparent node uses a New Micro's USB-XBEE-DONGLE CARRIER (the one under the shaped



SCREENSHOT 1. X-CTU and the USB-XBEE-DONGLE CARRIER are to the XBee-PRO module as MPLAB and an ICD-3 are to a PIC. X-CTU exposes all of the module's secrets and stores them away, as well.

glass and lights in **Photo 3**) to host an XBee-PRO module via one of the PC's USB ports. The CARRIER uses FTDI USB technology to interface the module's serial I/O to the PC's USB portal. If you're FTDI challenged, check back issues of *SERVO* and *Nuts & Volts* to get a snoot full of FTDI technical data and hands-on projects. You can get a free schematic of the CARRIER from the New Micro's website. The USB-XBEE-DONGLE CARRIER is powered by 5.0 volts obtained from the PC's USB port and provides onboard voltage regulation for the XBee-PRO module (which requires a nominal 3.3 volt power source). In addition to providing a regulated power source, the CARRIER allows us to configure the piggy-backed XBee-PRO module. The module's firmware can also be upgraded using the resources of the CARRIER coupled with the free XBee-PRO PC-based configuration application X-CTU. If you plan to work with XBee-PRO modules, the CARRIER is a must-have tool.

Screenshot 1 is a capture of an X-CTU modem configuration window. This view gives you an idea of some of the XBee-PRO knobs you can twist. For instance, the packetization timeout value for this particular XBee-PRO module is three character times — which equates to 312.5 μ s at 9600 bps. We can also change the data rate, parity setting, and enable API mode from this view. As you can see in the capture, X-CTU provides a terminal functionality which comes in handy if you don't happen to have a copy of Tera Term Pro loaded and ready to run.

Assembling and coding the PC node is a piece of cake. First of all, there is no real coding work to be done. Assembling the XBee-PRO hardware entails plugging an XBee-PRO radio into the CARRIER. Things aren't so straightforward on the PIC32MX795F512H side. The XBee-PRO is pinned at a 2.0 mm pitch, and our prototyping board is drilled at 0.1 inch centers. So, the first order of business is to adapt our module to the prototyping board.

Installing the XBee-PRO Radio Module

I know what you're thinking and yes, we could simply purchase an XBee-PRO adapter from a vendor on the Internet. The alternative is to dig into the stuff-I-didn't-use-for-that-other-project box and build an XBee-PRO adapter with sticks and rocks. Using my handy Dremel tool, I cut a 1.4" x 1.0" rectangle from a plated-through perfboard. The XBee-PRO module will fit into a pair of 10-pin 2.0 mm SIP sockets which I mounted on the perfboard using a quad of right-angle header pins. The 2.0 mm sockets are hovering over the perfboard and that gave me plenty of room to hand-wire their pins down to it (it's been fitted with two rows of standard 0.1 inch pitch header pins).

The completed caveman module adapter is shown in **Photo 4**. A 1.0 μ F ceramic capacitor paralleled with an 8.2 μ F ceramic capacitor is recommended to be electrically attached between pins 1 and 10 of the XBee-PRO module. The capacitors are mounted on the hidden side of the perfboard. The garage-brewed XBee-PRO module assembly

is mated to the prototyping board via matching 0.1 inch female header receptacles.

Dropping in the PIC

Okay. Now that we have a new radio for our ZeroG - PIC24FJ128GA006 Trainer, the "ZeroG" original development board name can now be considered "XBee-PRO." Our next modification will negate the development board's PIC moniker. Don't be tempted to mount a ZeroG module with the new 32-bit PIC installed, the SPI portal used by the ZeroG module is replaced by USB mechanics on the 32-bit PIC. Likewise, we should avoid utilizing the serial port that we installed over in *Nuts & Volts*. The NV serial port is tied to a dedicated ST3232 RS-232 voltage translator IC. If we choose to, we can still use the ST3232-equipped serial port to talk to other devices that need to see RS-232 voltage levels. However, the XBee-PRO will most likely produce some magic smoke if we try to connect its TTL serial interface to the ST3232's bi-level RS-232 interface. As you can see in **Photo 5**, the PIC32MX795F512H is laying the PIC24FJ128GA006's pad area like the egg of a brown-headed cowbird.

Doing the Paperwork

Schematics 1 and **2** document our engineering change activity. If you grab your copy of this month's *Nuts & Volts*, you'll notice that the PIC24FJ128GA006 in the Design Cycle discussion has been replaced here by a PIC32MX795F512H. Pretty much everything else remains the same with absolutely no changes made in **Schematic 2**. However, if you decide to use the ST3232-equipped serial port on the PIC32MX795F512H's RF4 and RF5, you'll need to code for UART3A instead of UART2. Note that the XBee-PRO module is feeding from the PIC32MX795F512H's UART2A tube.

As you can see in **Photo 6**, the hardware is assembled and ready to test. The schematics have also been updated. I've got my USB-XBEE-DONGLE CARRIER loaded with an XBee-PRO module, and a Tera Term Pro emulator session is running on my laptop. Let's see if we can push some characters out of the PIC32MX795F512H, through the module, and into the terminal emulator window.

Coding UART 2A

Microchip's MPLAB C for PIC32 MCUs supports a library of practical peripheral routines we can call upon for our UART2A test code. To obtain access to the library, all we need to do is include it in our source code:

XBee-Pro
Digi
www.digi.com

USB-XBEE-DONGLE CARRIER
New Micros
www.newmicros.com

PIC32MX795F512H
MPLAB C for PIC32 MCUs
Microchip
www.microchip.com

Sources



PHOTO 4. Here's my caveman module adapter. Note the point-to-point wiring for power, ground, serial I/O, and the active-low RESET pins.

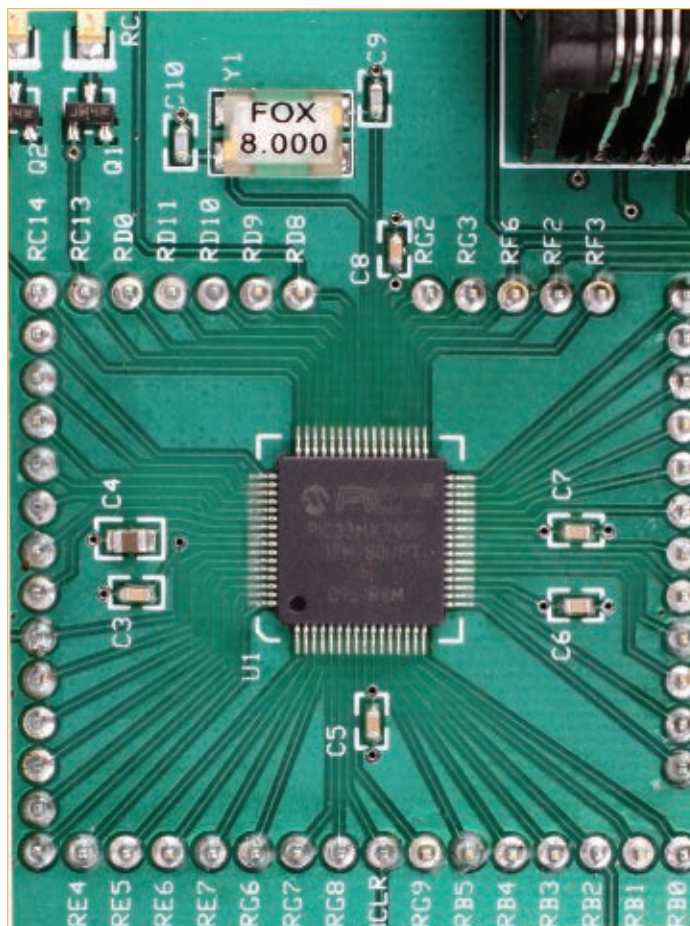
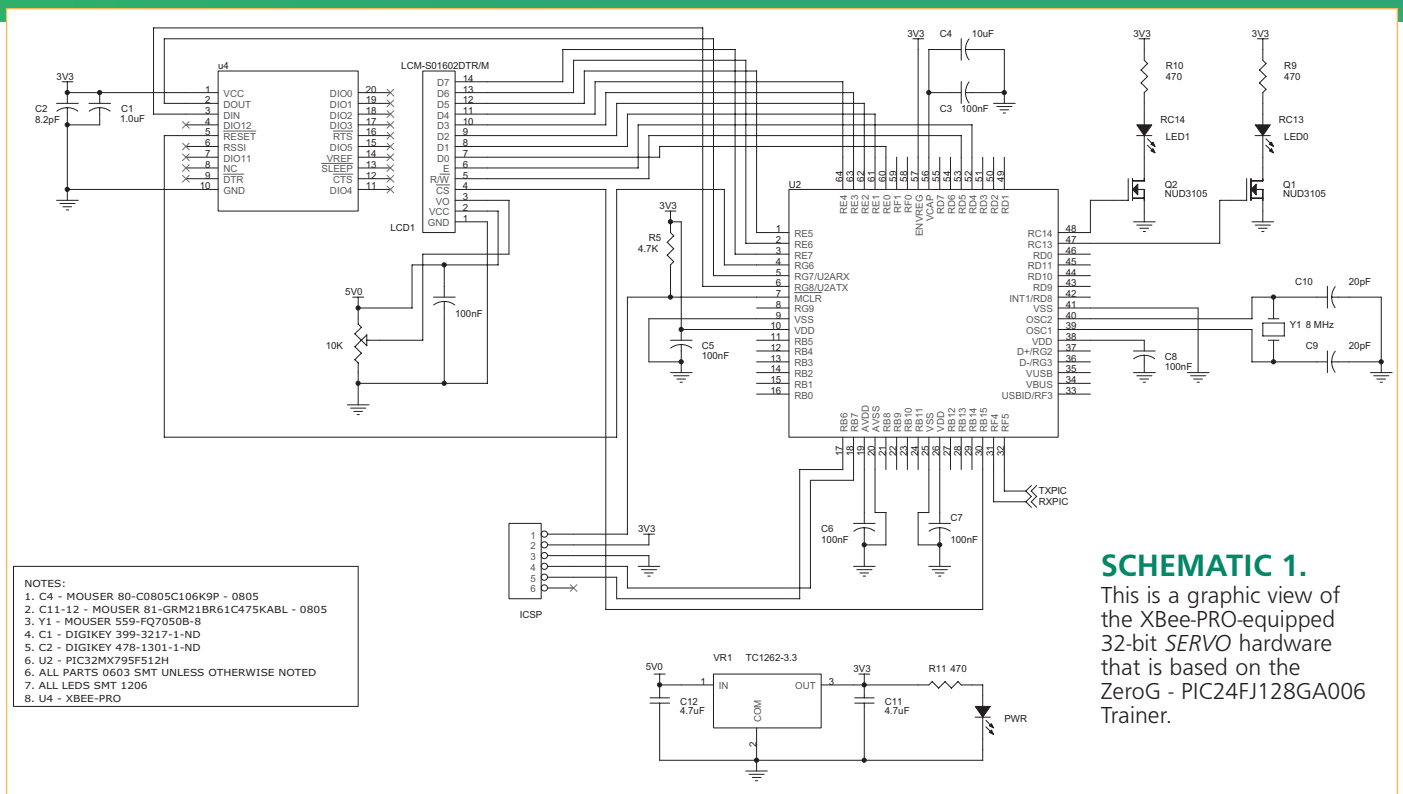


PHOTO 5. In addition to the PIC32MX795F512H, the PIC24FJ128GA106 and PIC24FJ128GB106 can also lie comfortably in this bed of pads.



SCHEMATIC 1.

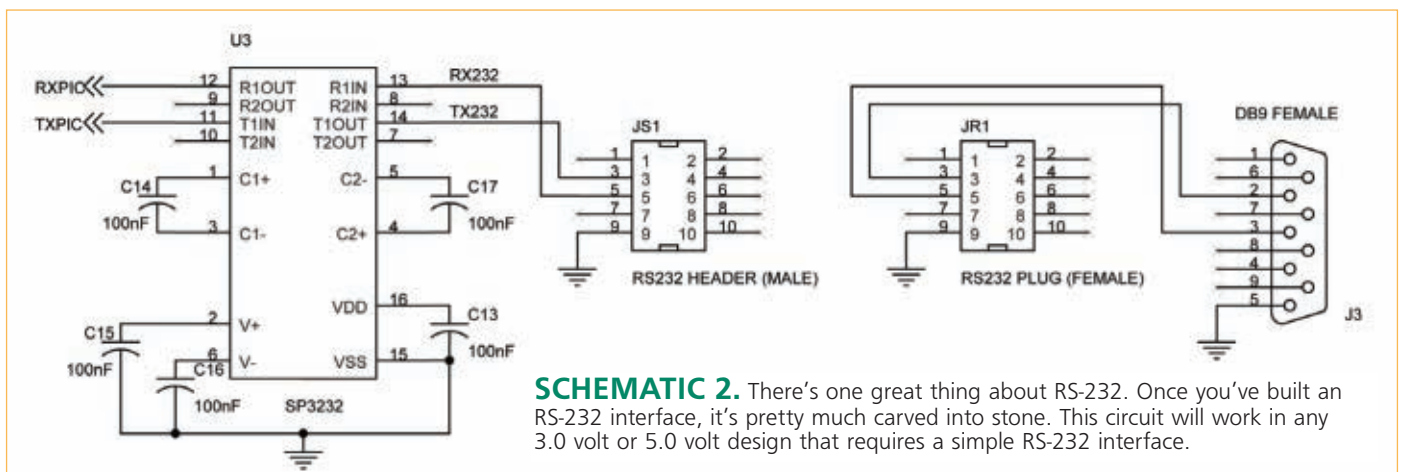
This is a graphic view of the XBee-PRO-equipped 32-bit *SERVO* hardware that is based on the ZeroG - PIC24FJ128GA006 Trainer.

```
#include <plib.h>
```

You can pick out the configuration fuses that interest you from the configuration fuse listing. However, for me, the most important fuses tell me how fast the PIC32MX795F512H's CPU is being clocked. Our hardware clock is derived from an 8.0 MHz crystal whose base frequency is multiplied by 20 on the way in and divided by two on the way out:

```
/** CONFIGURATION***** */
#pragma config UPLLEN
= ON // USB PLL Enabled
#pragma config FPLLMUL
= MUL_20 // PLL Multiplier
#pragma config UPLLDIV
= DIV_2 // USB PLL Input Divider
#pragma config FPLLDIV
= DIV_2 // PLL Input Divider
#pragma config FPLLODIV
= DIV_1 // PLL Output Divider
```

```
#pragma config FBPBIV
= DIV_1 // Peripheral Clock divisor
#pragma config FWDTEN
= OFF // Watchdog Timer
#pragma config WDTPS
= PS1 // Watchdog Timer Postscale
#pragma config FCKSM
= CSDCMD // Clock Switching & Fail Safe
#pragma config OSCIOFNC
= OFF // CLK0 Enable
#pragma config POSCMOD
= HS // Primary Oscillator
#pragma config IESO
= OFF // Internal/External Switch-over
#pragma config FSOSCEN
= OFF // Secondary Oscillator Enable
// (KLO was off)
#pragma config FNOSC
= PRIPLL // Oscillator Selection
#pragma config CP
= OFF // Code Protect
#pragma config BWP
```



SCHEMATIC 2. There's one great thing about RS-232. Once you've built an RS-232 interface, it's pretty much carved into stone. This circuit will work in any 3.0 volt or 5.0 volt design that requires a simple RS-232 interface.


```

= OFF      // Boot Flash Write Protect
#pragma config PWP
= OFF      // Program Flash Write Protect
#pragma config ICESEL
= ICS_PGx2 // ICE/ICD Comm Channel Select
#pragma config DEBUG
= ON       // Background Debugger Enable

```

Note also that the USB clocks are active in our configuration fuse listing. You never know when a USB connection may come in handy. The configuration PLL values have been chosen to yield a clock frequency that we can use to programmatically compute the register values for our desired baud rate of 9600 bps:

```

#define CLOCK_FREQ      80000000
#define DESIRED_BAUDRATE 9600

```

The baud register computation is embedded in a peripheral library UART call. So are the rest of the UART register fillers:

```

//CONFIGURE UART2A
UARTSetDataRate(UART2A,CLOCK_FREQ,
DESIRED_BAUDRATE);
UARTSetLineControl(UART2A,UART_DATA_SIZE_8_BITS
|UART_PARITY_NONE|UART_STOP_BITS_1);
UARTEnable(UART2A,UART_ENABLE_FLAGS(UART_ENABLE
|UART2A|UART_RX|UART_TX));

```

If you like to be in control, the plib calls are very uncomfortable. However, if you're into results without having to bang your head and flip through complicated user guides, the plib calls are heaven. Just reading the calls reveals everything that has been done to configure UART2A. I chose to drive the XBee-PRO module's active-low RESET line from a PIC32MX795F512H I/O pin (RG6). So, to make sure we're not placing the module into reset inadvertently, we must configure the I/O pin driving the module RESET input:

```

#define XBEE_RESET      LATGbits.LATG6
//XBEE RESET = 1
TRISGbits.TRISG6 = 0; //I/O pin = OUTPUT
XBEE_RESET = 1;      //XBee-PRO RESET disabled

```

This test code project is synonymous to a drag race. Most of the real work is done before the race and the race seems to be over as quickly as it starts. I hate it when I write a code snippet that is supposed to "talk" to me serially and then nothing seems to happen when I run the code. So, to make sure the code is actually running where I think it should be running, I like to flash a debug LED inside of the transmit routine. Here's the LED driver code:

```

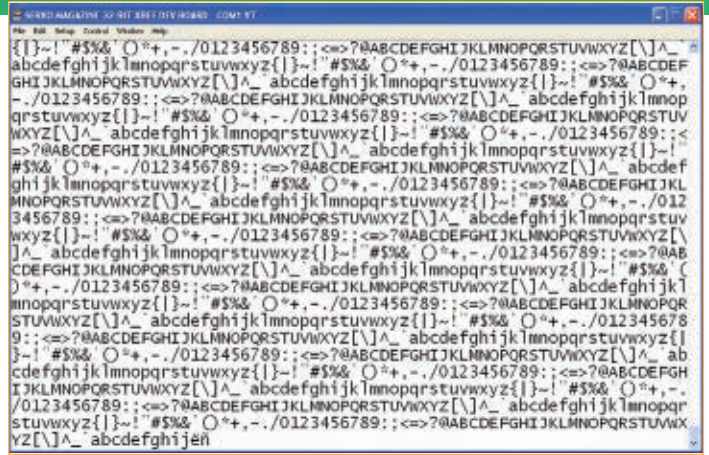
#define LED13          LATCbits.LATC13
#define LED14          LATCbits.LATC14
#define LED13_OFF()   LATCbits.LATC13 = 0
#define LED14_OFF()   LATCbits.LATC14 = 0
#define LED13_ON()    LATCbits.LATC13 = 1
#define LED14_ON()    LATCbits.LATC14 = 1
#define toggle_LED13() LED13 = !LED13
#define toggle_LED14() LED13 = !LED14

```

```

//SETUP LEDs
TRISCbits.TRISC13 = 0;
TRISCbits.TRISC14 = 0;

```



SCREENSHOT 2. This is a radio-to-radio capture.

The XBee-PRO module on our PIC32MX795F512H-ruled development system is transmitting characters between 0x20 and 0x7E to the USB-XBEE-DONGLE CARRIER hanging onto one of my laptop's USB ports.

```

LED13_OFF();
LED14_OFF();

```

After insuring that the PIC32MX795F512H LED I/O pins are configured as outputs, I turn off both LEDs. To blink the debug LED slowly enough for our eyes to discern the on/off states, we must kill time between switching the LED on and off. I've coded our UART squawk to transmit every displayable character between 0x20 and 0x7E. The code will always pass a 0x30 to the UART. So, every time a 0x30 (numeric zero) is transmitted, LED13 will change states. The code that performs the aforementioned tasks is, of course, obvious to the most casual programmer:

```

UINT8 rxdata;

rxdata = 0x20;
while(1)
{
    if (UARTTransmitterIsReady(UART2A))

```

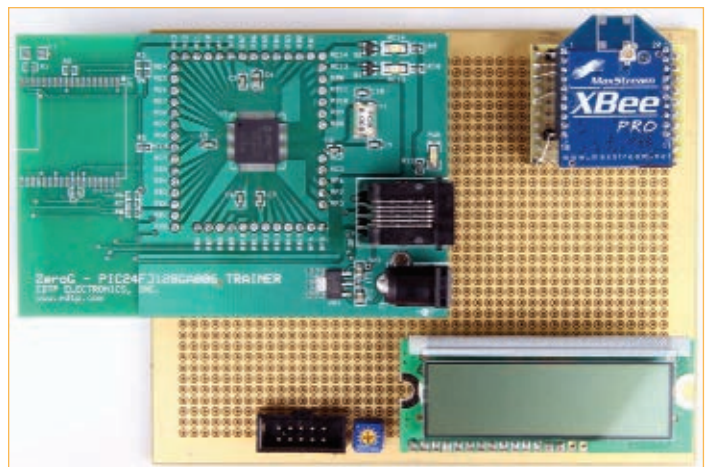


PHOTO 6. This project is dedicated to the readers that have asked me to tone down the costs of the projects. Here, we have a common prototype baseboard with a plug-in microcontroller carrier that utilizes pin-compatible 16-bit and 32-bit PICs.

```

{
    if(++rxdata == 0x30)
    {
        toggle_LED13();
    }
    if(rxdata > 0x19 && rxdata < 0x7F)
    {
        UARTSendDataByte(UART2A, rxdata);
    }
    else
    {
        rxdata = 0x20;
    }
}
}

```

The output of our UART test code is captured in **Screenshot 2**.

Eye Candy

There's no need to write the LCD code from scratch when we can "borrow" it from the TCP/IP stack. We'll use the time delay and LCD driver code that Microchip's Howard Schlunder used in the stack. Howard writes fine code, so, thanks!

To use Howard's stack code, we must emulate the stack's environment. For instance, it's a must to map the LCD pins to the PIC's I/O pins. You can check the mapping code against **Schematic 1**:

```

//*****
//**      LCD DEFINITIONS
//*****
#define LCD_DATA_TRIS (*(volatile BYTE*)&TRISE))

```

```

#define LCD_DATA_IO (*(volatile BYTE*)&LATE))
#define LCD_RD_WR_TRIS (TRISDbits.TRISD5)
#define LCD_RD_WR_IO (LATDbits.LATD5)
#define LCD_RS_TRIS (TRISBbits.TRISB15)
#define LCD_RS_IO (LATBbits.LATB15)
#define LCD_E_TRIS (TRISDbits.TRISD4)
#define LCD_E_IO (LATDbits.LATD4)

```

We also need to include Howard's timing routines:

```

#define GetSystemClock() (80000000ul) // Hz
#define GetInstructionClock() (GetSystemClock()/1)

//*****
//**      DELAY FUNCTIONS
//*****
void DelayMs(WORD ms)
{
    unsigned char i;
    while(ms--)
    {
        i=4;
        while(i--)
        {
            Delay10us(25);
        }
    }
}

void Delay10us(DWORD dwCount)
{
    volatile DWORD _dcnt;

    _dcnt = dwCount*((DWORD)(0.00001/
1.0/GetInstructionClock()/10));
    while(_dcnt--)
    {
        #if defined(__C32__)
            Nop();
            Nop();
            Nop();
        #endif
    }
}

```

The delays are used in the LCD initialization and operational functions and other parts of the application that need precision time delays. Here's a typical LCD banner code snippet:

```

BYTE LCDText[16*2+1];

LCDInit();
DelayMs(100);
strcpy((char*)LCDText,
        "SERVO MAGAZINE " //LCD LINE 1
        "XBEE-PRO PROJECT"); //LCD LINE 2
LCDUpdate();

```

Once the LCD is properly initialized, we simply write our message into the LCDText buffer which is allocated to hold two lines of 16 characters for the LCD. I've included the source code and instead of shooting a picture of the bench system, I'll let you put the two-line *SERVO* banner message on an LCD staring at you from your bench.

Now What??

That's easy. Interface, interface, interface. You have all of the tools you need to monitor sensors, display status via an LCD, activate relays, drive solid-state relays, and flash status LEDs. Plus, you can perform all the aforementioned tasks via the tip of a tiny wire antenna. **SV**



By Chris Savage

GPS Navigation Part 3

In this last part, I have refined the original test platform to be able to not only navigate from one waypoint to another, but to handle input from sensors and also to correct for errors in heading not adequately handled in the code from the last installment. This code is a bit more complex and so the focus will be on how the individual sections work to complete the task of navigation.

Code Tactics

Typically, there is a style to my programming that involves a main loop and many subroutines to complete the tasks of the program. Usually, the main loop is not much more than a series of subroutine calls. This usually makes it pretty easy to break the program up into manageable chunks. In this application, we'll be following a similar pattern although some of the subroutines will be handling tasks based on variables set in other subroutines. While not exactly unusual for BASIC programming, the logic may be quite different for controllers which pass parameters into or out of subroutines.

Upgrades

In order to handle the additional sensors, the original Super Carrier Board (**Figure 1**) was replaced with a new Super Carrier Board (**Figure 2**) with different connections for the various sensors used by the new code. There is also

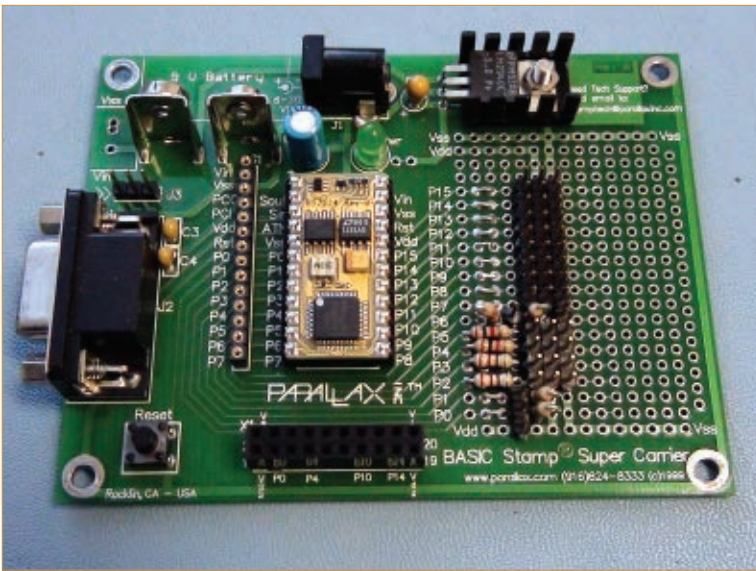


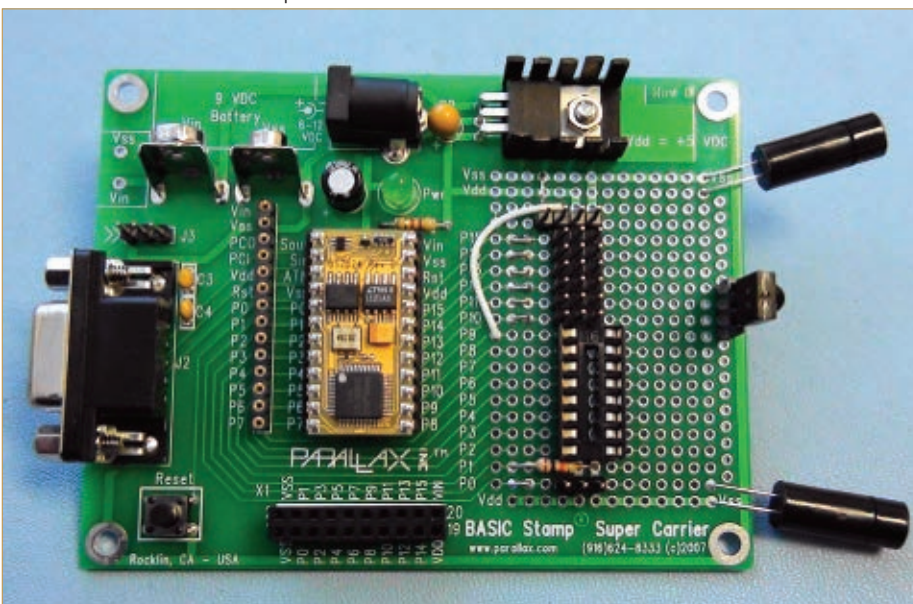
FIGURE 1. The old Super Carrier Board build.

some facility for expansion, although with only 16 I/O pins we're stretching things a bit now. The new board supports the following interfaces:

- Parallax Serial LCD
- Pushbutton ()
- IR Detector
- Multi-Channel ADC
- HM55B Compass Module
- (2) IR LEDs
- Parallax GPS Receiver
- (2) PING))) Sensors
- (2) HB-25 Motor Controllers
- (2) PING))) Mounting Bracket Kits

Both the ADC and the HM55B use an SPI-type interface meaning that the ADC only required one additional I/O pin.

FIGURE 2. The new Super Carrier Board build.



The IR emitters are configured to use only one detector. The GPS module is configured for raw output mode by tying the /RAW pin to V_{SS} . The PING))) modules are set up to detect a car in the path of the robot, which is why they are positioned on top of the robot. The HM55B compass module is moved up and away from the PCB, which sits directly over the two NiCd batteries and four motors.

Revised Programming

Once the changes to the control board were completed, it was time to download the revised program code onto it. The revised program has a main loop that does the following tasks until the last waypoint has been reached:

- Check for obstacles (handle avoidance).
- Obtain target GPS coordinates.
- Obtain current GPS coordinates.
- Calculate desired heading based on target position vs. current position.
- Obtain current heading from compass module.
- Adjust course based on desired heading vs. current heading.
- Handle heading change > 90 degrees.
- Determine if waypoint has been reached (and is the last one).
- Adjust course and speed of motor drive as needed to reach goal.

This loop will continue until the robot reaches its goal or the batteries die. Once the robot has reached its destination, the LCD will display a confirmation message that the course is complete.

Code Details

The first task now is to check for obstacles. This adds a certain factor of complexity in that the sensor code is trying to adjust your course away from an obstacle while the navigation code is trying to keep you on your current heading. The solution is to override the navigation routines and move out of the way first. After moving some distance, the code can then check to see if it can get to the desired coordinates from its new position. Unfortunately, with the current sensors coming in at the wrong angle it can cause the robot to not see surfaces it is approaching from a low angle. As **Figure 3** shows, some obstacles will require some fine-tuning to the sensors.

After obstacles have been checked for, we obtain the coordinates of the current position. These coordinates are

used along with the target coordinates to calculate the desired heading. As before, we can also get the distance to target in this section. Next, we want to obtain the current heading from the compass module and compare it to our desired heading. Any course corrections are applied by the motion routines which also handle ramping and speed changes. The skid-steering design is quite nice for this type of application, however, a tail-wheel robot with differential drive would be just as effective. The ability to turn in place has advantages in this type of application.

Final Thoughts

To really appreciate this project, you need to see it in action. To that end, videos of this version of the GPS Navigating robot will be available (see **Resources sidebar**). As with many projects of this type, development will be ongoing until I am happy with the performance of the design given the available resources. Eventually, this robot

Resources	Parallax, Inc. www.parallax.com
	Savage Circuits YouTube Channel www.youtube.com/savagecircuits
	Project Page www.savagecircuits.com/gpsnav/



FIGURE 3. Certain obstacles proved to be a challenge.

chassis will pave the way for a Stingray version of the code which will, of course, run on a Propeller, but will also add additional sensors including wheel encoders.

Having just obtained some Windows 7 compatible flowchart software, I should have a flowchart for the code up on the project page. The flowchart will be very detailed about what the code is doing. It is my hope that this will make it easier to port the routines over to your favorite microcontroller. Remember, the code is heavily commented to help, as well.

As I wrap up this article, I want to remind you about my website. Besides hosting all my projects and videos, the site is forum-based. If you prefer videos, most projects have video tutorials. If you have any feedback, feel free to contact me (info@savagecircuits.com). I hope you'll be able to apply all these directions to your projects. **SV**



AP CIRCUITS
PCB Fabrication Since 1984

As low as...

\$9.95
each!

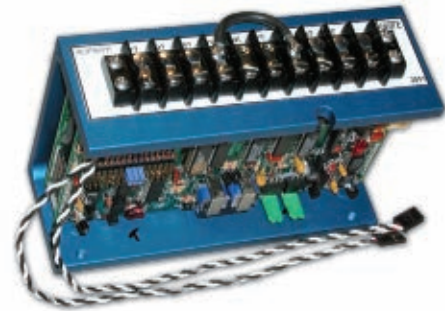
Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com



STEER WINNING ROBOTS WITHOUT SERVOS!



Perform proportional speed, direction, and steering with only two Radio/Control channels for vehicles using two separate brush-type electric motors mounted right and left with our **mixing RDFR dual speed control**. Used in many successful competitive robots. Single joystick operation: up goes straight ahead, down is reverse. Pure right or left twirls vehicle as motors turn opposite directions. In between stick positions completely proportional. Plugs in like a servo to your Futaba, JR, Hitec, or similar radio. Compatible with gyro steering stabilization. Various volt and amp sizes available. The RDFR47E 55V 75A per motor unit pictured above.
www.vantec.com

VANTEC

**Order at
(888) 929-5055**

The **SERVO** Webstore

Attention Subscribers ask about your discount on prices marked with an *

CD-ROM SPECIALS



6 CD-ROMs & Hat Special
Only \$ 129.95 (includes shipping)!

www.servomagazine.com



On Sale Now!
Only \$24.95

ROBOTICS

PIC Robotics by John Iovine

Here's everything the robotics hobbyist needs to harness the power of the PICMicro MCU!



In this heavily-illustrated resource, author John Iovine provides plans and complete parts lists for 11 easy-to-build robots each with a PICMicro "brain." The expertly written coverage of the PIC Basic Computer makes programming a snap – and lots of fun.

\$24.95

FIRST Robots: Rack 'N' Roll: Behind the Design

by Vince Wilczynski,
Stephanie Slezyski

More than 750 photographs!

The second annual book highlighting the creativity and process behind 30 winning robot designs from the 18th annual international FIRST Robotics Competition. The FIRST organization, founded by Dean Kamen (inventor of the Segway), promotes education in the sciences, technology, and engineering.

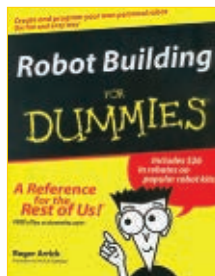
Reg \$39.95



Robot Building for Dummies by Roger Arrick / Nancy Stevenson

Discover what robots can do and how they work. Find out how to build your own robot and program it to perform tasks. Ready to enter the robot world? This book is your passport! It walks you through building your very own little metal assistant from a kit, dressing it up, giving it a brain, programming it to do things, even making it talk. Along the way, you'll gather some tidbits about robot history, enthusiasts' groups, and more.

\$24.95



Build Your Own Humanoid Robots

by Karl Williams

GREAT 'DROIDS, INDEED!

This unique guide to sophisticated robotics projects brings humanoid robot construction home to the hobbyist. Written by a well-known figure in the robotics community, *Build Your Own Humanoid Robots* provides step-by-step directions for six exciting projects, each costing less than \$300. Together, they form the essential ingredients for making your own humanoid robot. **\$24.95***



Robot Programmer's Bonanza by John Blankenship, Samuel Mishal

The first hands-on programming guide for today's robot hobbyist!

Get ready to reach into your programming toolbox and control a robot like never before! *Robot Programmer's Bonanza* is the one-stop guide for everyone from robot novices to advanced hobbyists who are ready to go beyond just building robots and start programming them to perform useful tasks.

\$29.95



Robotics Demystified

by Edwin Wise

YOU DON'T NEED ARTIFICIAL INTELLIGENCE TO LEARN ROBOTICS!

Now anyone with an interest in robotics can gain a deeper understanding – without formal training, unlimited time, or a genius IQ. In *Robotics Demystified*, expert robot builder and author Edwin Wise provides an effective and totally painless way to learn about the technologies used to build robots! **\$19.95**



We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

To order call 1-800-783-4624

SERVO Magazine Bundles



Published by T & L Publications, Inc.

\$57
per bundle

Save \$10
off the
normal
price!!

Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, 08, and 09.

Kickin' Bot

by Grant Imahara

Enter the arena of the metal gladiators!

Do you have what it takes to build a battle-ready robot? You do now! Here are the plans, step-by-step directions, and expert advice that will put you in competition — while you have a heck of a lot of fun getting there. Grant Imahara, the creator of the popular BattleBot Deadblow, shares everything he's learned about robot design, tools, and techniques for metal working and the parts you need and where to get them.

\$24.95



How Would You Like To See Your House Cleaned?



TODAY



SERVO's VISION of the FUTURE

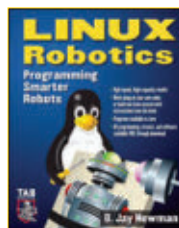
Visit my online store today! <http://store.servomagazine.com>

Linux Robotics

by D. Jay Newman

If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need.

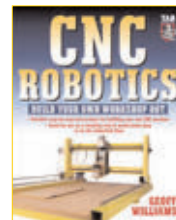
\$34.95



CNC Robotics

by Geoff Williams

Here's the FIRST book to offer step-by-step guidelines that walk the reader through the entire process of building a CNC (Computer Numerical Control) machine from start to finish. Using inexpensive, off-the-shelf parts, readers can build CNC machines with true industrial shop applications such as machining, routing, and cutting — at a fraction of what it would cost to purchase one. Great for anyone who wants to automate a task in their home shop or small business. **\$34.95**



SPECIAL OFFERS

The Amateur Scientist 3.0 The Complete Collection

by Bright Science, LLC

There are 1,000 projects on this CD, not to mention the additional technical info and bonus features. It doesn't matter if you're a complete novice looking to do their first science fair project or a super tech-head gadget freak; there are enough projects on the single CD-ROM to keep you and 50 of your friends busy for a lifetime!

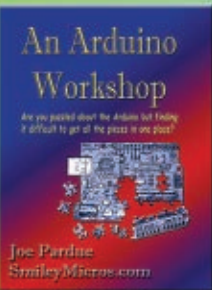
\$29.95



The Getting Started Combo includes: *Getting Started in Electronics* by author Forrest Mims and the DIY Electronics Kit. In his book, Mims teaches you the basics and takes you on a tour of analog and digital components. He explains how they work and shows you how they can be combined for various applications. The DIY Electronics Kit allows for the hands-on experience of putting circuits together — the kit has over 130 parts! No soldering is required and it includes its own 32 page illustrated manual.

Combo Price \$62.95 Plus S/H

SPECIAL OFFERS



Book \$44.95

Puzzled by the Arduino?

Based on the Nuts&Volts Smileys Workshop, this set gives you all the pieces you need!

Book and Kit Combo \$124.95

Kit 84.95

For more info on this and other great combos!
Please visit: <http://store.nutsvolts.com>



Combo Price \$175.95

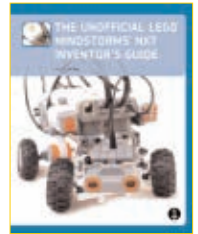
Enter the world of PICs & Programming with this great combo!

For complete details visit our website @ www.nutsvolts.com

The Unofficial LEGO MIND-STORMS NXT Inventor's Guide

by David J. Perdue

This book was written for the first version of the NXT set (#8527), and its projects are only compatible with the first version. In other words, because of piece differences between the NXT 1.0 and 2.0 sets, the projects in this book can only be built with an NXT 1.0 set. However, much of the other information is still helpful. Much of the building, mechanical, and programming information is applicable. **Reg \$29.95**
Sale Price \$25.95



Forbidden LEGO

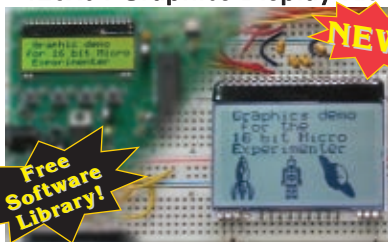
by Ulrik Pilegaard / Mike Dooley

Forbidden LEGO introduces you to the type of free-style building that LEGO's master builders do for fun in the back room. Using LEGO bricks in combination with common household materials (from rubber bands and glue to plastic spoons and ping-pong balls) along with some very unorthodox building techniques, you'll learn to create working models that LEGO would never endorse. **Reg \$24.95** **Sale Price \$19.95**



PROJECTS

128x64 Graphics Display Kit



New application for the 16-Bit Micro Experimenter

LCD displays ... they have been around for quite some time, but what if you could have both characters as well as graphic displays at the same time? With this kit, we will show you how easy and inexpensive this technology can be using the 16-Bit Micro Experimenter.

Subscriber's Price \$45.95
Non-Subscriber's Price \$48.95

16-Bit Micro Experimenter Board

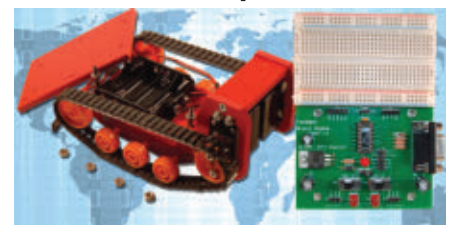


Ready to move on from eight-bit to 16-bit microcontrollers? Well, you're in luck! In the December 2009 Nuts & Volts issue, you're introduced to the 16-Bit Micro Experimenter.

The kit comes with a CD-ROM that contains details on assembly, operation, as well as an assortment of ready-made applications. New applications will be added in upcoming months.

Subscriber's Price \$55.95
Non-Subscriber's Price \$59.95

Tankbot Kit & Brain Alpha Kit



Tankbot/Brain Alpha
originally by Ron Hackett
Now with New Columnist Calvin Turzillo
A series filled with projects and experiments to challenge you through your learning process while you grow your fully expandable Brain Alpha PCB!
The brain is a PICAXE-14A!

For more info & pictures, visit the *SERVO* Website.
Tankbot and the Brain Alpha Kit can be purchased separately.

Combo Price \$ 138.95

5th ANNUAL

Maker Faire®

THE
World's
Largest
DIY Festival

A two-day, family-friendly event to **MAKE**, create, learn, invent, **CRAFT**, recycle, think, play, celebrate, and be inspired by arts, crafts, engineering, food, music, science, and technology.

Rockets ■ Robots ■ Art Cars ■ Eepy Bird Diet Coke & Mentos Fountain Show
Exploratorium ■ Cyclecide ■ Life-Size Mouse Trap ■ Swap-O-Rama-Rama
Bazaar Bizarre Craft Fair ■ Food Makers ■ Sustainable Living ■ And more!



SO MUCH
TO SEE,
YOU NEED
2 DAYS
TO SEE
IT ALL!



Advance tickets,
subscription
deals, & more
information
**available
online! »**

May 22 & 23, 2010

SAN MATEO COUNTY EVENT CENTER

Saturday 10am–8pm / Sunday 10am–6pm

MakerFaire.com

BROUGHT TO YOU
BY MAKE MAGAZINE



FREE
ADULT
DAY PASS
(\$25 value)
when you
subscribe
to MAKE.

makerfairetickets.com

Brought to you by **MAKE:** magazine

Using a VEX Controller

Numeric LED Display Experiment

Continuing with experiments that can be carried out using a VEX microcontroller, this article describes how you can build a DIY numeric LED display that can provide a bright, easy to read display for VEX-based projects. Now that you know how to use the VEX microcontroller from the first installment in the Mar '10 issue, we'll pick up the pace a bit to show more advanced numeric LEDs and how to drive large numbers of discrete LEDs while at the same time conserving battery power and precious VEX I/O pins, reducing the overall cost of external components.

Numeric Display LEDs have been around for many years and provide a bright display that can be seen in both bright daylight and at night time. They were first used for electronic equipment and consumer appliances in the '70s, including pocket calculators such as the HP-65 and TI-59. Although low power LCD displays and OLEDs have replaced them in new calculators and commercial appliances, they still continue to show up in special test equipment and DIY hardware. When used in VEX applications, they can provide sensor data such as motor RPM readings, VEX ultrasonic ranger readings, and encoder readings that are visible from a distance under various lighting conditions (including total darkness).

A numeric LED digit is composed of seven LED segments that come in two basic configurations: cathode and anode. For our experiment here, I used the common cathode version since it requires less parts. I also selected a MAXIM (www.maxim.com) MAX7219 numeric LED display driver to drive up to five cathode digits as shown in the schematic in **Figure 1**. Although the cost of this IC is a bit high (around \$8), in my opinion it is well worth it — considering how it simplifies the hardware and firmware for VEX-based projects. This is because the MAX7219 IC provides all the necessary timing and refreshing to display numbers using bright LEDs. The +5 volt power supply necessary to drive all the numeric digits is supplied directly

from one of the microcontroller's +5 volt pins (middle) located on the Analog/Digital I/O block. The MAX7219 IC can be cascaded to drive even more numeric LEDs as needed, although the control and selection logic will get more complicated, and will use more I/O pins on the VEX controller. The interface requires six signals to correctly download the commands and data to the MAX7219 controller IC. Numeric LEDs are soldered to the breadboard to complete the display assembly. The completed circuit board with the display showing the value 31415 is shown in **Figure 2**. The MAX7219 can also display decimal points for floating point numbers.

MAX7219 numeric LED commands that can be sent to the controller include: changing the brightness of each digit; changing the numeric value of each digit; changing the refresh rate; turning the decimal point on or off; and running the test mode that turns on all the segments from each digit by displaying the number 8 (see **Table 1**).

These commands are then sent to the MAX7219 using a serial protocol. This is described in great detail in the datasheet located at (<http://datasheets.maxim-ic.com/en/ds/MAX7219-MAX7221.pdf>). The MAX7219 LED display board shown in **Figure 2** can also drive directly up to 64 LEDs individually when connected in a similar manner to each numeric LED segment. This feature allows VEX users to add lighting to their robots.

For one of my own numeric LED applications (the DIY

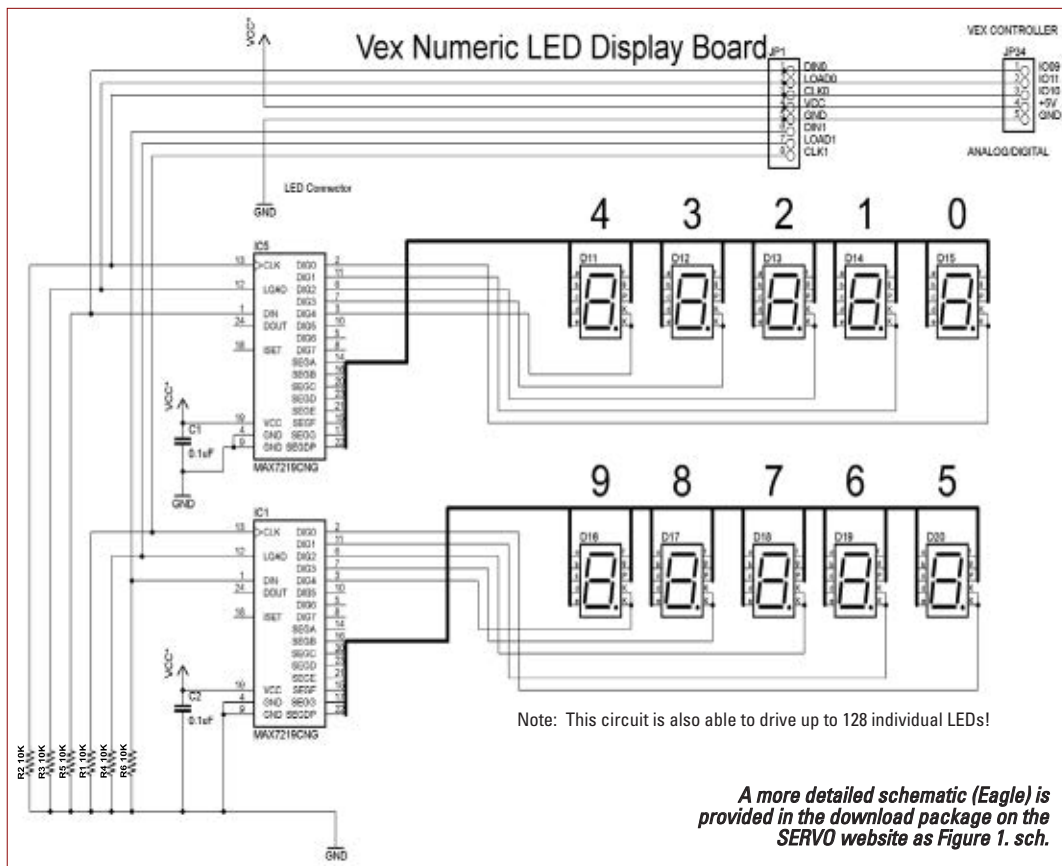


FIGURE 1. For this experiment, I used a MAX7219 numeric LED display driver to manipulate five numeric LED displays as shown in this schematic.

that I can use the MAX7219 to add lighting to the passenger cabin and the main engine so that it looks cool when it runs at night.)

TABLE 1. MAX7219 Register Address Map.		
Function	Hex Code	Description
No-Op	0x0	No Operation performed
Digit 0	0x1	Address of digit 0
Digit 1	0x2	Address of digit 1
Digit 2	0x3	Address of digit 2
Digit 3	0x4	Address of digit 3
Digit 4	0x5	Address of digit 4
Digit 5	0x6	Address of digit 5
Digit 6	0x7	Address of digit 6
Digit 7	0x8	Address of digit 7
Decode Mode	0x9	Decode register; a 1 turns on BCD decoding
Intensity	0xA	Intensity register
Scan Limit	0xB	Scan-limit register
Shutdown	0xC	On/Off register
Display Test	0xF	Activates test mode (all digits on, 100% bright)

PIC18 C Numeric LED Display Application

The C code that drives the numeric LED display is located in the file max7219.c which is provided along with

FIGURE 2. The completed numeric LED display circuit board.





FIGURE 3. My DIY HP-45 calculator using two MAX7219 ICs to make this colorful 10-digit numeric LED display by just wire-wrapping each LED segment to the MAX7219 driver.

all the necessary source code files on the *SERVO* website at www.servomagazine.com. In **Listing 1**, I show you how to send the necessary configuration commands to the MAX7219 using the information provided in **Table 1** and how to display the value of PI. There is also a second example showing you how to use the driver routines to display integer values ranging from 0 to 99999 in **Listing 2**. This example calls routine *itoa* to first convert the integer value to an ASCII string and then calls *LED_Print* to send each digit to the MAX7219 LED driver so that it can be displayed.

The code was developed using Microchip MPLAB and the PIC18 C compiler; it also uses the WPILIB library. The code configures the controller's digital ports to outputs and sets each LED segment to the desired state by issuing the appropriate MAX7219 command. In the code, you see an infinite while loop that converts the integer value to ASCII and sends each digit to the MAX7219 so that it can display the appropriate value from 0 to 99999. It then waits for a few milliseconds before displaying the next digit.

In the source code, I included functions to initialize the MAX7219 LED controller and to send the characters 0 to 9; ".", and "H," "E," "L," "P" using the *LED_Print* function. To include support for your own applications, just add

ROBOTC is another C-language programming environment specifically targeted for the VEX products. It is developed by the Robotics Academy at Carnegie-Mellon University. It includes features like interactive single step debugging with variable watching and real-time monitoring of sensor values and motor settings. A 30-day evaluation version is available from www.robotc.net. It does require the orange colored programming cable that comes with the VEX Programming Hardware Kit.

(Test) Tool Time

Here are some ideas for test equipment to help you with troubleshooting.



Voltmeter – A basic diagnostic tool for debugging circuits is a digital voltmeter (DVM). The DVM shown in **Figure A** can be used to measure quantities such as voltage, current, resistance, capacitance, inductance, and continuity at various test points in your circuits. The continuity tester beeps if there is a good electrical connection between any two points. Use a schematic as a guide and test all the connections systematically.

Logic probe – A low cost digital probe indicates a digital I/O pin is high or low, or pulsing using a simple LED display. Logic probes typically support both TTL and CMOS digital outputs.

Oscilloscope – This is a very useful electronic diagnostic and test tool, although some models can be very expensive. It provides a beam trace of one or more analog signals on a CRT or LCD display at various frequencies ranging from 20 MHz to 1 GHz. Oscilloscopes can be used to measure voltage, frequency, amplitude, period, current, and many other signal properties. Dual trace oscilloscopes can be found for around \$70 to \$200.

Logic Analyzer – This digital diagnostic tool displays the state of digital signals, busses, etc. They can easily display eight, 16, 32, or 64 traces at one time to show the timing relationships between signals. They store the data in a very large buffer so that the signals can be captured and played back later for analysis.

Virtual Instruments – This is PC or laptop based software that simulates oscilloscopes, logic analyzers, and digital voltmeters. This is one way to keep costs down, but still get the data you need.

Why Use a MAX7219 LED Display Driver?

Numeric LED displays can be driven directly from the VEX controller using discrete components instead of the MAX7219 IC. However, but to do this requires complicated firmware to pulse each LED at the selected refresh rate using timers and Interrupt Service Routines (ISRs), in addition to using most of the controller's I/O pins unless a decoder IC is also used. That's why I chose the MAX7219.


```

char TestNumber1[] = "3.14159263";

//*****
//* main - main function
//*****

void main(void)
{
    int i;

    // Configure USART for 40 MHz,
    // 9600 Baud, 8-Bits, No parity,
    // Asynchronous
    InitializeUsart();

    printf("\r\n VEX MAX7219 LED
    Controller \r\n");

    // Setup the timers used for delays.
    SetupTimers();

    // Initialize the MAX7219 Numeric LED
    // Display Driver
    InitializeMAX7219();

    // Display the value of PI
    LED_Print(TestNumber1);

    while(1);
}

```

LISTING 1.

```

// Display digitis from 0 to 99999 on the
// Numeric LED Display using the MAX7219

while(1)
{
    for (i=0; i<99999; i++)
    {
        // Convert the integer i to an ASCII
        // string
        itoa(i, TestNumber);

        // Display the string on the Numeric LED
        // Display using the MAX7219.
        LED_Print(TestNumber);

        // Wait 10 milliseconds
        pause(10);
    }
}

```

LISTING 2.

"#include max7219.h" to your VEX C programs and max7219.c to your MPLAB project files. Feel free to use the code and modify it as you see fit.

Programming the VEX Controller

It's time to download the max7219.hex application into the VEX controller and run it. Start by copying the led.hex file to your laptop or PC hard disk and place it in a folder. Program the VEX microcontroller's Flash by running the IFI Bootloader (also available from the IFI site) and then browse to the directory you created for the max7219.hex file. Once

TABLE 2. Bill of Materials (BOM) for the Numeric LED Display Board.

ITEM	QTY	DESCRIPTION	SOURCE
1	1-2	MAX7219 Numeric LED Driver IC	Jameco.com
2	1	Prototype Breadboard	RadioShack
3	1-10	Common Cathode Numeric LEDs	Digi-Key.com
4	1	VEX Microcontroller	IFI
5	3-6	10K Resistors	RadioShack
6	1-2	.1 µF Capacitors	RadioShack
7	100	.100 Pin Headers (optional)	Digi-Key
8	1	Wire-wrap Wire (optional)	RadioShack
9	1	Wire-wrap Tool (optional)	RadioShack

TABLE 2. The total quantities of each component depend on how many MAX7219 ICs are used (one or two). If you plan to use point-to-point construction, then there is no need to purchase the wire-wrap materials.

the max7219.hex file has been downloaded successfully, the display should show the value 31415.

If the display does not work correctly the first time you power it up, don't panic. It could just be a or loose wire or even a bad electronic component. What you need to do is double-check all the wiring against the schematic with the circuit disconnected from the power supply. Check the +5 volts and Ground connections, and also the Vdd, Vcc, and Vss and GND pins of each IC to make sure the power supply wiring is correct.

Applications for the Numeric LED Display

There are many VEX related uses for numeric LED displays including digital displays for timers, clocks, thermometers, speedometers, FRC field scoring displays, and battery voltage displays. When integrated with a keypad, they can be used for combination locks, calculators, and motor controllers. You can also use the MAX7219 for driving up to 64 LEDs which can be attached to robots and props to brighten them up while using less power.

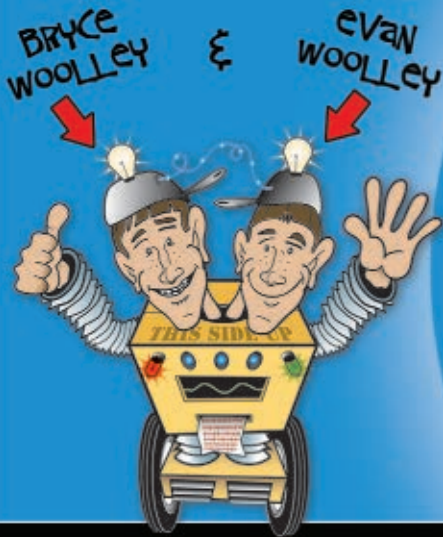
The Fun Starts Here

Now that you have some preliminary information under your belt with some exposure to VEX, its logic subsystem, and the various internal and external electronic components (including the MAX7219 IC and the numeric LED display), you can begin to integrate discrete LED and numeric LED displays into your own VEX projects.

If you come up with a really cool application, be sure to let *SERVO* know. **SV**

Twin Tweaks

THIS MONTH:



*Robonova –
Come Here –
I Want to
See You*



PRINCE MYSHKIN,
RETURNED FROM SECLUSION.

This month, we have the privilege to present the VeeAR VRbot voice recognition module which comes to us courtesy of Tegal. Tegal is a high-tech electronics distributor from Austria, and the VRbot module is designed for use with the Robonova-1 from Hitec (and with its Japanese cousin, the Robozak). Given our previous trials and travails with the Robonova, we thought perhaps a good talking to might bring it around.

Debugged in the Sanitarium

Way back in the May and July '06 issues of *SERVO*, we had the opportunity to work with the Robonova-1 robot from Hitec. To celebrate the Tetsujin 2006 competition, we outfitted the Robonova with an exoskeleton designed to take on the cylinder stacking challenge. Unfortunately, our Robonova's last adventure ended with a shameful denouement that earned him the moniker of Prince Myshkin. The hip servo in his left leg went crazy, kicking out and refusing to cooperate when it was supposed to be in the zero position. While this was likely just a problem with the potentiometer, we were still inclined to name him after Dostoevsky's epileptic hero. And

just like his literary namesake, our ill-fated Robonova has spent its time since then in seclusion, though not anywhere as swank as a Swiss resort.

The VRbot module has given us an excellent opportunity to retrieve the Robonova from its hiding place. Our first order of business was to address the ailment that banished the robot in the first place: Myshkin's faulty hip. We ordered an extra HSR-8498HB servo and set about the surgery. The cleverly modular design of the Robonova made it so that we did not have to extensively disassemble the bot to get to the troublesome servo. The surgery also gave us the opportunity to replace some of the tiny screws that hold the whole bot together. All of the Robonova's acrobatics had shaken loose a few of

its screws back in the day, and unfortunately the extra parts included in the kit did not quite meet our needs. Thankfully, **SmallParts.com** came to the rescue, and finding the unusually tiny machine screws (PH/T-2 2 x 4) was a snap.

With all of the necessary parts, replacing Myshkin's hip simply became a matter of dexterous fingertips. Once the hapless Prince was back in one piece, we were thrilled to see him stand tall instead

THE VEEAR VRBOT MODULE FROM TEGAL



of assuming the penalty kick position when turned on.

With Prince Myshkin fully recovered, we were ready to wire up the VRbot module. Maybe with voice command capabilities Myshkin could go from the idiot to part of Craig Ferguson's Robot Skeleton Army. There was only one way to find out ...

Eye of the Tigel

The VRbot module can be acquired from the Tigel website for a cool 39 euros, but included in that price tag is a polished little unit and a plethora of support from the website. The website includes all of the documentation that a tinkerer could ever wish for, with everything from datasheets to software.

The VRbot module itself is a breeze to hook up with only three connections to be made. One is a socket for the microphone, and the other two wires connect to the Robonova itself. The Tigel website features an illuminating diagram that shows exactly where to plug the wires into the Robonova. The board on the Robonova itself (the MR-C3024) clearly labels the pins (VCC, ground, and serial transmit and receive) to dispel any second-guessing.

The VRbot board is small enough to mount inside of the Robonova body panels, which seems to us to be quite a necessity given the rough and tumble acrobatics of the bot. An instructional video on the Tigel website that goes step by step through the process of implementing the VRbot module recommends affixing the board to the inside of the front panel of the Robonova using double-sided tape. The wires can actually run through the center of the bot's chest to reach the pins, and we also found that the back body panel even has enough room to house the board. After

closing up the robot, the only evidence of its new abilities was the small microphone popping out the body panel. In true Van Gogh fashion, we attached Myshkin's new ear on the right side of his head with a bit of tape. Now we were ready for some voice training.

A nice feature about the VRbot module is that it does not need any extra connection to the computer to be programmed. The board comes with a bridge program that simply piggybacks off of the connection between the computer and the Robonova microcontroller board. As mentioned in the instructional video, though, the VRbot module can be connected directly to the computer with a serial adapter.

Much to our often mentioned chagrin, the Robonova (and, by extension, the VRbot module) connects to the computer using an RS-232 serial cable. The VRbot module folks must have foreseen the headaches caused by the insistence on using this obnoxiously persistent cable, and in one of the text files that come with the VRbot GUI they make recommendations for some compatible serial-to-USB adapters. Our trusty Keyspan adapter did not make it onto the list, but it worked just fine. With the connection sorted out, we needed to get our hands on the proper software.

The Tigel website provides the VRbot GUI for programming. The instructions available on the website are also readable and include a multitude of helpful screenshots. The instructions point out and label the relevant icons on the GUI, which is an explanation that often seems overlooked in graphical interfaces. To the developers, the meaning of the icon that looks like a green meat grinder might be obvious, but it is helpful to have a translation in the manual. The mysterious meat grinder was actually a serial cable icon with a green box over it, and the button connects the robot to the computer when clicked. The VRbot

*PRINCE MYSHKIN'S HIP REPLACEMENT:
AN HSR-8498HB SERVO FROM HITEC.*



HIP SURGERY FOR MYSHKIN.



Twin Tweaks ...



THE VRBOT MODULE READY FOR INSTALLATION.



WIRING UP THE VRBOT MODULE.



PRINCE MYSHKIN'S NEW EAR.

GUI is refreshingly easy to use, and the Robonova is ready to listen after the first connection.

Pygmalion for Robots

The VRbot module comes with a delightful bridge program that downloads to the Robonova after the first connection to the GUI program. The bridge program includes a number of so-called speaker independent commands that allow for immediate functionality. Essentially, the program includes triggers and wordsets. The default trigger word is "robot." Saying the trigger word puts the robot into a sort of listening mode where it is ready for a specific command. The bridge program cleverly uses the LEDs in the Robonova's head to indicate when the Robonova has received commands. After saying "robot," the LED will turn on. Then you can give commands like "Hello," which will cause the robot to bow. Other commands like "move" or "attack" will cause the LED to flash. This means that the Robonova is awaiting further instructions. Further instructions like "forward" or "right" will cause the Robonova to take a few steps in that direction or to launch a flurry of punches and some fancy footwork.

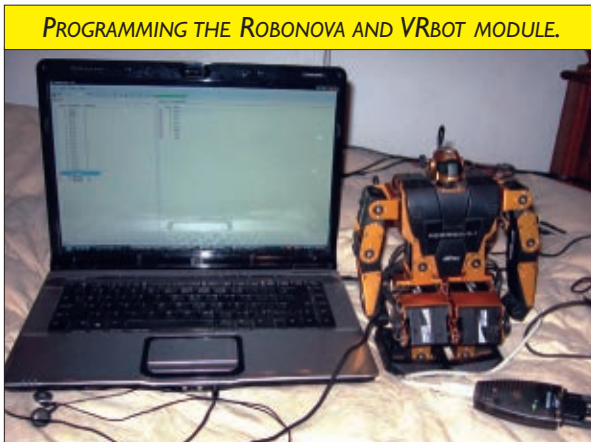
The GUI has a way for the user to test the recognition of the voice commands with the "test group" function. While in testing mode, the

program prompts the user to say a command. If the Robonova picks up on the word, it is highlighted by a green flashing box in the GUI. While this might seem like a great opportunity for you to test your diction about the rain in Spain, the test function is a great way to test the sensitivity of the microphone.

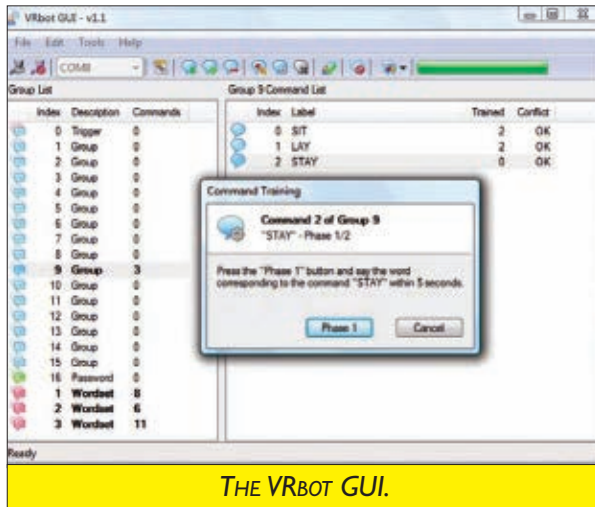
We started out with our best public speaking voices, sure to pass muster even with the discerning ear of Professor Higgins. Sure enough, this also passed muster with the microphone, and words like robot, move, and eight lit up without fail. Even with a conversational tone, the microphone had no problem picking up on our words.

Of course, we realize that not all users will share our conversational enunciation, and we were wondering how much variation the microphone could tolerate. Our findings were encouraging. We discovered that the VRbot module can pick up on everything from British accents to whispers to voices like the Pillsbury Doughboy. Oddly enough, the microphone seemed to pick up on the Pillsbury Doughboy better than the British accent, but the mic didn't seem to have too much trouble as long as the voice was more Michael Caine than Ozzy Osbourne. We also tried our hand at a Spanish accent, and the VRbot module did fine as long as we didn't become too overenthusiastic when it came to rolling our Rs.

On a more practical note, we also used the test program to get an idea of the range of the microphone. All of our initial tests were with the Robonova seated next to the computer, and we hoped that the bot's auditory integrity would not decay dramatically with distance. We moved the Robonova to a nearby desk, and the GUI still consistently highlighted the correct words. We tried speaking softly, and the GUI still identified our commands accurately. We then banished the Robonova to the corner of the room — as far as the programming cable would reach. We were thrilled to discover that the microphone still picked



PROGRAMMING THE ROBONOVA AND VRBOT MODULE.



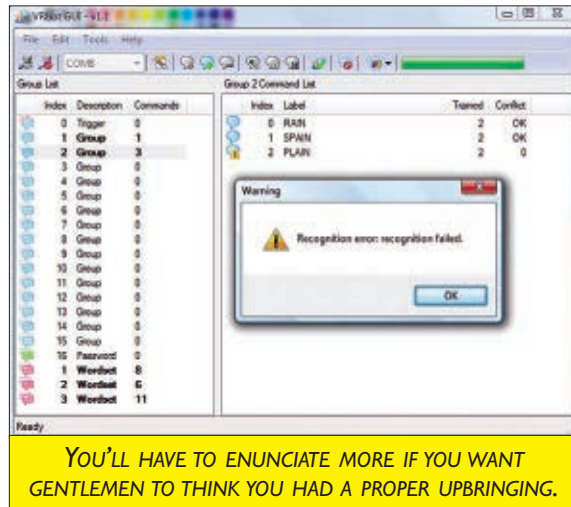
THE VRBOT GUI.

up our commands, even at a conversational volume. We pushed our luck by trying to whisper and seeing if the bot could hear. Understandably, it preferred us to speak up, but even at a decent distance the microphone could pick up a loud whisper.

The real fun comes with programming custom voice commands, and the fairly intuitive process is once again detailed in the instruction video and in a PDF file on the Tigel website. After connecting the robot to the computer, simply select the command group that you want to edit. Using the buttons on the toolbar, you can choose to add a command to the group. After naming the command something descriptive, the next step is to train it. Training comes in two phases where the user must simply say the command. After training, the test group function is a great way to see that your efforts worked.

You can even give your robot a name that it responds to by programming a new trigger word. We, of course, chose to train the name that we had already given the Robonova, though we decided against pretentiousness and dropped the title so that we would simply call the bot by "Myshkin."

As we were messing around with different voice commands, we wondered how the VRbot module would deal with similar sounding words. Fortunately, the developers have thought of that very problem, and we tested their solution in true Pygmalion fashion. First, we trained the command "rain," and we predictably did so without incident. Next was the command "Spain," which we also trained without a problem. Finally, we trained the word "plain," and this brought up a warning. A dialog box cautioned us that this was similar to one of our other trained commands. The command was flagged with a yellow sign, and we proceeded to test the group with some

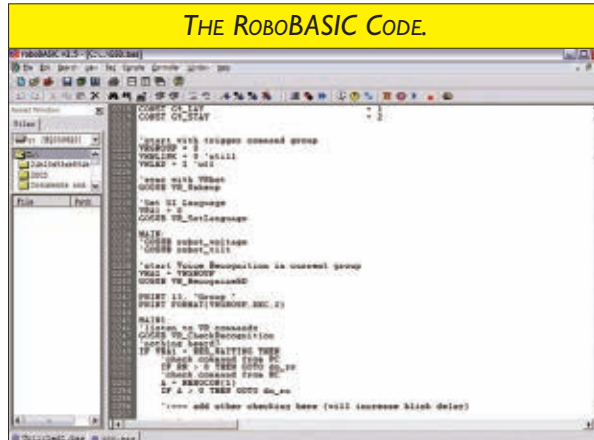


YOU'LL HAVE TO ENUNCIATE MORE IF YOU WANT GENTLEMEN TO THINK YOU HAD A PROPER UPBRINGING.

trepidation. With careful diction, we recited the apparent truism about Spanish weather, and we were pleased to see that all the words were recognized correctly. The next time through we let ourselves get a little sloppy, and the test was unable to distinguish between the words "rain" and "plain." Trying again with our best Queen's English, the program recognized all of the words. So, it seems that users can train their Robonova in limericks if they so desire, but they must simply be prepared to speak clearly. Professor Higgins would be proud.

After being thoroughly impressed by the capabilities of the VRbot GUI, we thought it was time to put our voice training to the test. What would be a real test of Myshkin's obedience? The obvious thing to do was to match up the bot against someone else that had received similar training – our German Shepherd dog Argus. Argus, in true Deutscher Schäferhund fashion, has been trained to "sitz" (sit), "platz" (lay down), and "blieb" (stay). Learning commands in German is quite all right for Myshkin, because the multicultural VRbot module supports commands in English, German, Japanese, and Italian.

THE ROBObASIC CODE.



Twin Tweaks ...



*NOW MYSHKIN CAN ATTACK
ON COMMAND!*



*THOUGH THE METHODS OF
TRAINING MIGHT DIFFER ...*



THE RESULTS ARE THE SAME!

We trained our German commands, but then had to implement them in code. The VRbot GUI has one last nifty feature: a generate code function that will spit out a basic file that can be modified to attach the desired movements to the trained commands. All we had to do now was implement the code.

Hooked on Phonics

Even though we recently polished up our BASIC programming skills with the COSMOS kinetic sculptures (Jan '10 issue), we were a bit rusty with the Robonova syntax. Unfortunately, implementing the generated code was one place that the instructional video glossed over by simply urging the user to modify the code according to their needs. The PDF file and the code itself provide some clues. The PDF manual directs the user to certain lines of code, and the .bas file includes some helpful comments. We were still hoping for a bit more help, and we were delighted to discover that there is a very helpful forum on the VeeAR website. One of the threads on the forum was about our very conundrum of implementing the code generated by the GUI. The thread provided sample code and detailed instructions about where to include it. We felt like we were home free – just a bit of copying and pasting, and we would be ready for download!

Not quite. We fired up the RoboBASIC editor and were distressed to find that our generated basic file would not open. We tried starting a new program file that we could just copy the code into, and that was unavailing as well. We looked to the Hitec website for answers, and we were somewhat relieved to find a file proclaiming itself to be a RoboBASIC fix for Windows Vista. We duly downloaded the software and ran it, but RoboBASIC remained obstinately closed for business. Taking another look through the VeeAR forums, we saw several recommendations to use the updated version of

RoboBASIC (version 2.72). Another look at the Hitec website revealed only links to download the older 2.5 version, which is the same release that came on our original Robonova CD. Once again, the VeeAR forums came to the rescue, and directed us straight to the source – the RoboBASIC website. The RoboBASIC website allowed us to download the new version 2.72, and we were confident that this would solve all of our problems.

Not quite. The newer version still refused to open our .bas file, or even to start a new one. Thinking the problem was rooted in the computer and not the software, we switched to a different laptop, this one running Windows 7. We knew this was a gamble, and it was one that didn't pay off. RoboBASIC persisted in its unusual behavior, and we were afraid that Prince Myshkin might never be appointed with his new name or vocabulary.

As a last ditch effort, we brought one of our old laptops out of retirement to see if RoboBASIC would get along better with trusty old Windows XP. We held our breath as we opened our basic file, and we were delighted to see that the window burst open with many splendorous lines of code.

We wouldn't want to rule out the efficacy of the Vista fix based on our experience alone, but the moral of the story seems to be one of caution when using technology of questionable compatibility. If you don't have an old operating system on a computer with an RS-232 serial port, be prepared with an adapter (we recommend Keyspan) and a backup plan.

Obedience Training

After finally outfitting Prince Myshkin with

RECOMMENDED WEBSITES

www.tigal.com
www.veear.eu/Forums.aspx
www.robobasic.com

his new voice commands, we were ready to test his obedience against that of Argus. After calling the attention of the students with their respective names, we issued the command "Sitz!" Argus responded quickly, but Myshkin required a few repetitions before finally acquiescing. In his defense, Myshkin had certainly received less training than Argus, and he did eventually comply. With that sort of obedience, a spot on Ferguson's Skeleton Army might not be out of the question.

Overall, we were very pleased with the accessibility and effectiveness of the VRbot module. Even after working with countless robot kits, there is something unusually exciting about seeing a robot respond to your commands — not by a remote control but by the sound of your voice. The LEDs are a super helpful debugging tool, as is the test group function in the GUI. And while the microphone seemed more sensitive in the test group mode than our little obedience class, we're sure that finding the right conversational speed with the Robonova would make things go much smoother. Some of the unresponsiveness may also have been due to our aged battery pack,

because Prince Myshkin did seem the most responsive after just being charged. So, while he'll keep the moniker of Prince Myshkin, at least it's a name that the bot can actually respond to now.

The excitement of the voice commands also got us thinking about what else we could use the VRbot module for. While the module is advertised to work with the Robonova, we think that implementation with other platforms is not only a possibility but something that is indeed encouraged by the folks at Tigel and VeeAR. The Tigel page concludes the instructional video with a few minutes on how to implement the module with the EasyPIC5 development board, and the VeeAR forum is rife with advice about using the module with different microcontrollers. With the right amount of ingenuity and coding savvy, we're sure that intrepid tinkerers could get just about any BASIC programmed robot to listen to them like an obedient German Shepherd. Now that deserves a treat. **SV**

SPECIAL THANKS TO

Markku Riihonen

NEW! Orangutan SVP-1284 Robot Controller

Item #1327 & #1328
\$99.95 and \$89.95



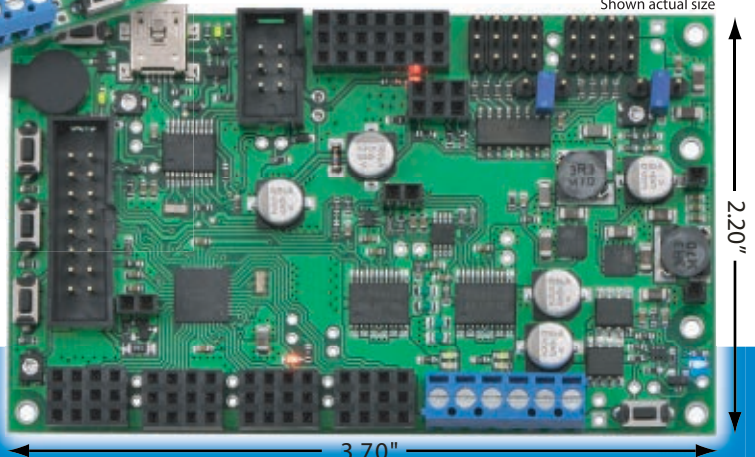
The **Orangutan SVP-1284** is available fully-assembled and as a kit.

The **Orangutan SVP-1284** lets you program your robot in C/C++ with free AVR development software. Use Pololu's AVR C/C++ library to easily make your **motors** go, **servos** turn, **LCD** print, and **buzzer** play - the Orangutan has all of these hardware features (and more) built right in!

Key Features:

- * **Wide input voltage range** - 6-13.5 V.
- * **C/C++ programmable microcontroller** - User-programmable Atmel ATmega1284PA AVR running at 20 MHz (128 KB flash, 16 KB SRAM, 4KB EEPROM).
- * **Built-in USB AVR ISP programmer** (USB A to mini-B cable included).
- * **Two bidirectional motor ports** - 2 A continuous/6 A maximum per channel.
- * **8 servo ports** - Use Pololu's AVR C/C++ library to control up to 8 servos!
- * **21 free I/O lines** - Up to 12 can be analog inputs, optional dual quadrature encoder inputs, two hardware UARTs.
- * **Removable LCD** - 16-character x 2-line, with backlight.
- * **Two step-down voltage regulators** - Each capable of supplying 3A.
- * **Other fun stuff** - Buzzer, 3 user pushbuttons, 2 user LEDs, and more!

Shown actual size



Pololu
Robotics & Electronics
3095 Patrick Lane #12, Las Vegas, NV 89120

Save 10%
with coupon code
SRV3SVP7

Learn more at www.pololu.com/svp or call 1-877-7-POLOLU



Then and NOW

WHAT DOES A ROBOT LOOK LIKE?

b y T o m C a r r o l l

In the mid '90s, I was assisting with some robot demos at the California Museum of Science and Industry in Los Angeles for kids at the museum's two-week robot camp. A few of the children had physical motor disabilities, but one young girl was blind, only having had sight for the first few years of her young life. She asked me one question which has given me a lot to think about over the years ... "What does a robot look like?"

What does a robot look like? What *should* a robot look like? Does a robot *have* to look a particular way to be considered a robot?

These are seemingly simple but very complex questions to ponder. Does a robot resemble a roving pie tin like the i-Robot Roomba vacuum cleaner? Are robots huge arms spitting sparks and paint like those used in a car factory? Are they small two pound creations with a wedge on the front trying to knock a similar robot out of a Sumo ring? Should they be humanoids like C-3PO of *Star Wars*?

It's doubtful that the average kid (or adult) would be able to recognize the first industrial robot — the Unimate in **Figure 1** from the early '60s. To most people,

this robot looked more like a tank turret on a box with an extendable arm instead of a gun. Yet, it revolutionized manufacturing the world over.

Rossum's Universal Robots

Most people today have become educated enough to realize that the robots of science fiction and the movies are not what exists in reality. However, they still do not know what has been developed over the past 90 years. Karel Capek's

early play, "RUR" actually introduced the world to the robots in 1920. Karel wrote the play, but his brother, Josef, actually coined the word robot from the Czech word 'robota' which means "drudgery" or "servitude." Capek's Robots were more like biomechanical beings than modern electro-mechanical creations. **Figure 2** shows a display model of the robot from the 1923 New York production of *RUR*. As you might have noticed, this robot is not a suit worn by an actor (since the hip joints cannot fit over a person's legs).

The play was written in 1920, premiered in Capek's hometown of Prague, Czechoslovakia in 1921, and then debuted in New York in 1922. This unique play changed

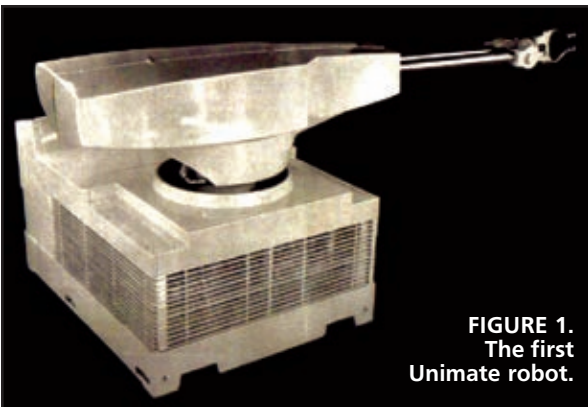


FIGURE 1.
The first
Unimate robot.

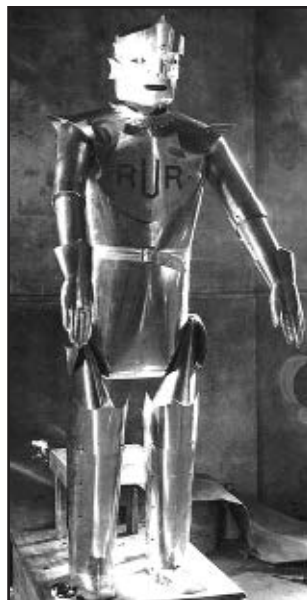


FIGURE 2. The
1923 RUR robot
mockup.



FIGURE 3. Golem from Jewish folklore.



FIGURE 4. The Stepford Wives from the 1975 movie.



FIGURE 5. *Revenge of the Nerds* robot.

how people viewed mechanical servants. Of course, it took George Devol and Joe Engelberger (in the 1960s) to actually build the first industrial robot — and it looked nothing like the robots of earlier times.

How do Humans Depict Robots?

For use in plays or movies actors can depict robots by dressing up as stiff-walking humanoids (like the Golem of old Jewish folklore shown in **Figure 3**) or appear brain-dead and beautiful like the *Stepford Wives* from the 1975 film (see **Figure 4**).

Another way to portray a robot is to actually make a *real* robot. **Figure 5** shows a robot I built for the first *Revenge of the Nerds* movie. (It reminds me of a rolling salt shaker with arms.) Of course, you can always have a person walk around in a robot suit like the beautifully-crafted Robbie from *Forbidden Planet*. Or you can design one on a computer and duplicate enough to fill a room as was done in the movie *I-Robot*. **Figure 6** shows the NS-5 robots surrounding Will Smith.

Building Your Idea of a Robot

When planning to build a robot, many people first decide on what they want the robot to do, then consider the aesthetics later on. ‘Human touches’ come a bit later after the robot construction is in progress. Line-following, combat, or Sumo robots built for these contests generally take a certain form conducive to winning the particular event. Adding any special appearance may be just a paint job or a few LEDs. People who build robots to roam around their homes or neighborhoods usually add touches such as dual arms, eyes, a mouth, and a humanoid body shape. Why? Because no matter what, that’s what

people expect a robot to look like.

Ask a young kid to draw a person and the result will be a round head atop a body usually with of two legs and two arms like in **Figure 7**. Ask the same kid to draw a robot and the result may be something like in **Figure 8** or **Figure 9**.

Humanoid Robots Take Center Stage

Humanoid robots have always been in the minds of robot designers, even before movies that featured robots. The technology escaped the average robot experimenter until the advent of walking technology and balancing techniques that used newly-developed accelerometers and multi-axis gyros. Honda’s Asimo and Korea’s Hubo have taken center stage for the human-sized humanoid robots, but another robot has captured world attention: Reem B, built in 2008.

Created by Pal Technology Robotics based in Abu Dhabi, United Arab Emirates, the 132 pound robot stands a bit under five feet tall. Though the company is based in the UAE, the team consists mostly of Spanish scientists working in Barcelona, Spain and is led by Davide Faconti, 29, who is Italian. This prototype follows an earlier REEM-A



FIGURE 6. NS-5 robots with Will Smith.

What Does a Robot Look Like?

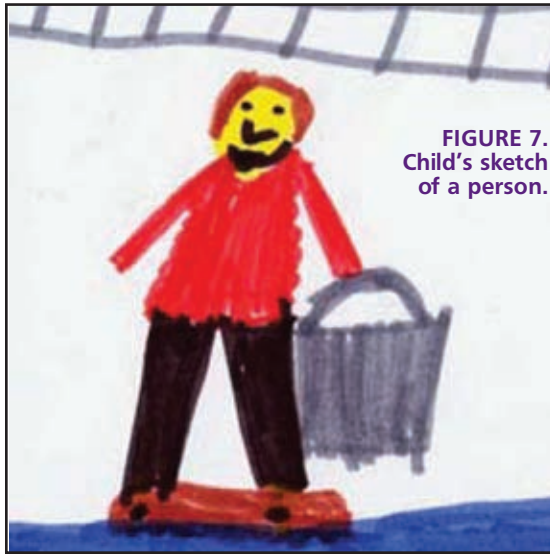


FIGURE 7.
Child's sketch
of a person.

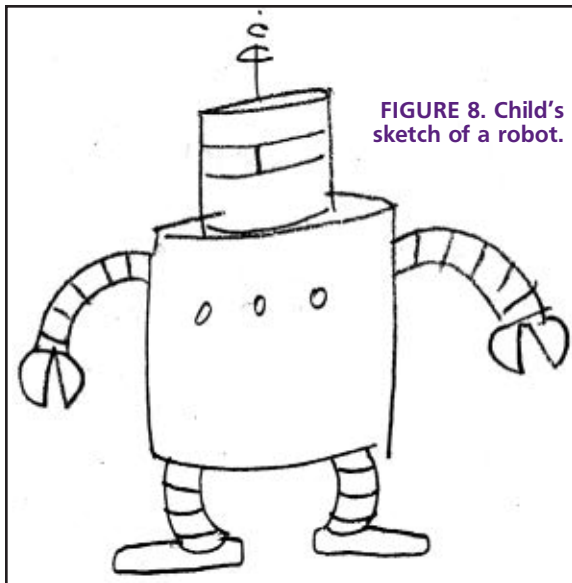


FIGURE 8. Child's
sketch of a robot.

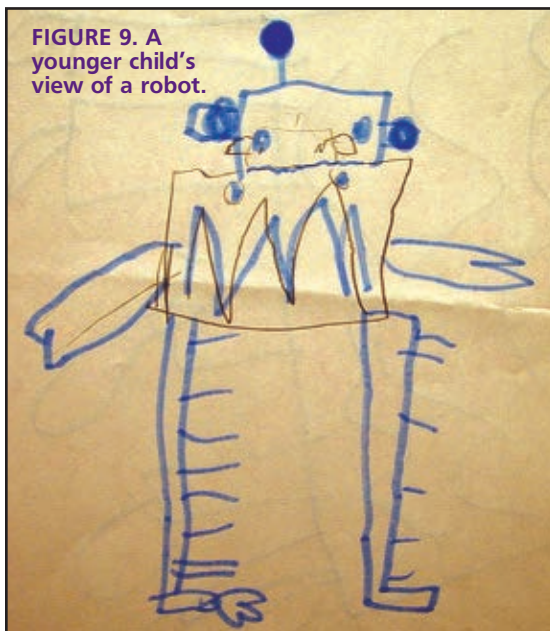


FIGURE 9. A
younger child's
view of a robot.

built in 2006. **Figure 10** shows Sheikh Mohamed bin Zayed of the UAE with REEM-B and Noel Sharkey, Professor of AI and robotics at the University of Sheffield, UK. This robot has some amazing capabilities that overshadow its two Asian counterparts. With 41 degrees of freedom, the ability to carry 25% of its weight, and operate for two hours are amazing enough. REEM-B also has advanced facial, object, and speech recognition, and highly-developed articulation and article grasping capabilities.

Final Thoughts

So, how should a robot look? It should look like what *you* want it to look like. It's *your* creation. And, like they say, robot beauty is in the eye of the builder. **SV**

A unique exhibit — *Robots: Evolution of a Cultural Icon* — examines the development of robot iconography in fine art over the past 50 years. The image and the idea of a robot has evolved remarkably from an awkward, mechanical creature to a sophisticated android with artificial intelligence and the potential for human-like consciousness. As robotic technology catches up with the imagination of science fiction novels, movies, and animation, dreams and fears anticipated in these stories may also become reality. Artists included in the exhibition have responded to the technological innovation with optimism, pessimism, and humor. The exhibit is running at the Boise Art Museum now through May 16, 2010. For more info, go to <http://boiseartmuseum.org/exhibit/current.php>.

*Tom Carroll can be reached at
TWCarroll@aol.com.*

FIGURE 10. REEM-B with Sheikh Mohamed and Professor Sharkey.



ROBO-LINKS

GPS MADE SIMPLE™



Linx Technologies.com

BiPOM Electronics, Inc. Your One Source for μ Controller Systems
ARM, AVR, PIC, 8051, 68XX, STAMP and others

μ Controller Boards
Peripherals
Development Tools
Emulators
Programmers
Robotics

GadgetPC ARM9 USB
Computer with Linux

Sensors
Dev. Training Kits
Instruments
Displays

WebCat+
Embedded Web Server

www.bipom.com

RobotShop.com



THE ORIGINAL SINCE 1994

PCB-POOL®
Beta LAYOUT

- Low Cost PCB prototypes
- Free laser SMT stencil with all Proto orders

WWW.PCB-POOL.COM

Pololu Robotics & Electronics
3095 E. Patrick Ln. #12, Las Vegas, NV 89120
www.POLOLU.com 1-877-7-POLOLU

Motor, servo, and robot controllers
Sensors

Robots, motors, wheels, and more!



\$29.95 (MSRP) MaxSonar-EZ1
High Performance Ultrasonic Range Finder

- serial, analog voltage & pulse width outputs
- lowest power - 2mA
- narrow beam
- very easy to use!

www.maxbotix.com



The most comprehensive resource for news and links

THE ROBOT REPORT
TRACKING THE BUSINESS OF ROBOTICS

www.TheRobotReport.com

For the finest in robots, parts, and services, go to www.servomagazine.com and click on Robo-Links.

NUBOTICS™
www.nubotics.com
Distributed by Acroname.com

LINEAR SERVOS

Firgelli
www.firgelli.com

ALL ELECTRONICS CORPORATION
Electronic Parts & Supplies Since 1967



Nuts & Volts 6 CD-ROMs & Hat Special!
That's 72 issues. Complete with supporting code and media files.

NUTS VOLTS 2004
NUTS VOLTS 2005
NUTS VOLTS 2006
NUTS VOLTS 2007
NUTS VOLTS 2008
NUTS VOLTS 2009

Free Shipping!
Only \$129.95



5 CD-ROMs & Hat Special
Only \$ 109.00 (includes shipping)!

www.servomagazine.com



ADVERTISER INDEX

All Electronics Corp.21, 81	Lynxmotion, Inc.83	RobotShop, Inc81, 82
AP Circuits63	Maker Faire67	Solarbotics/HVW53
BaneBots48	Maxbotix81	SparkFun ElectronicsBack Cover
BiPOM Electronics81	Nu-botics21, 81	The Robot Report81
Critical Velocity21	PCB Pool60, 81	Vantec63
Firgelli21, 81	Pololu Robotics & Electronics ..77, 81	WeirdStuff Warehouse21
Linx Technologies81	Robotis2	Yost Engineering3



Robots, lots of robots!

The World's Leading Source
for Domestic and Professional Robot Technology



www.robotshop.com



The Lynxmotion Servo Erector Set Imagine it... Build it... Control it!

Featured Robot

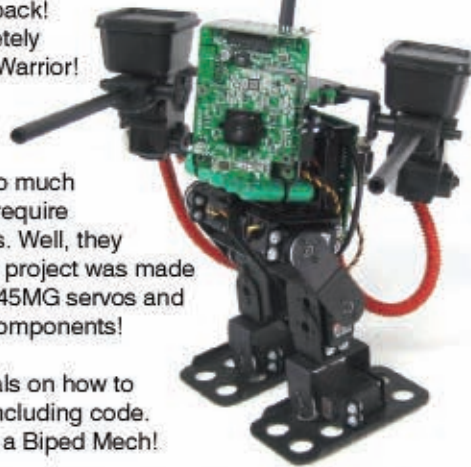
Biped BRAT does Mech Warfare!

Introducing the Hunchback!
The world's first completely functional biped Mech Warrior!

They said a BRAT based Mech couldn't be done. That it was too much payload. That it would require expensive digital servos. Well, they were wrong! This robot project was made from a BRAT with HS-645MG servos and standard off the shelf components!

We have created tutorials on how to make a Mech Warrior including code. Now anyone can make a Biped Mech!

Youtube videos
User: Robots7



Biped Nick



Biped Pete



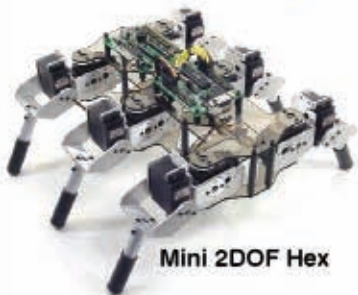
Biped Scout



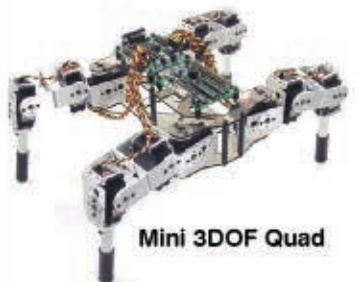
Biped 209



Walking Stick



Mini 2DOF Hex



Mini 3DOF Quad

With our popular Servo Erector Set you can easily build and control the robot of your dreams!

Our interchangeable aluminum brackets, hubs, and tubing make the ultimate in precision mechanical assemblies possible. The images here are just a sample of what can be built. The Bot Board II and SSC-32 provide powerful control options. Our Visual Sequencer program provides powerful PC, Basic Atom, or BS2 based control.



Bot Board II - \$24.95
Carrier for Atom / Pro, BS2, etc.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
Buffered Speaker.
Sony PS2 game controller port.
3 Push button / LED interface.



SSC-32 - \$39.95
32 Channel Servo Controller.
Speed, Timed, or Group moves.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
TTL or RS-232 Serial Comms.
No better SSC value anywhere!

We also carry motors, wheels, hubs, batteries, chargers, servos, sensors, RC radios, pillow blocks, hardware, etc!



Visit our huge website to see our complete line of Aluminum and Lexan based robot kits, electronics, and mechanical components.



CH3-R Hexapod



Biped BRATs



Phoenix



Images represent a fraction of what can be made!

www.lynxmotion.com

The SES now has 157 unique components!

Let your geek shine.

Meet Dwight Eppinger, SparkFun customer and Interactive Marketing Manager at Colorado's Copper Mountain Ski Resort. Using SparkFun's LED matrices and XBee modules, Dwight created a status board that updates the resort trail map signs on the condition of ski runs. From one computer, Dwight can instantly let ski patrol, Copper Mountain staff, and the people zipping down the slopes know which trails to hit.

Whether you're building a status system for an entire mountain or just wirelessly reaching across the room, the tools are out there. Find a new way to communicate, and let your geek shine too.



Sharing Ingenuity
WWW.SPARKFUN.COM

©2010 SparkFun Electronics, Inc. All rights reserved. All other trademarks contained herein are the property of their respective owners. Get the scoop on Dwight's project at crossingwiresarduino.blogspot.com. You can also read more about Copper Mountain Resort at www.coppercolorado.com.